

# Chương 01

# PHP Căn Bản

# Phân 01 – Xin chào PHP !

## Câu hỏi 1: PHP là gì ?

- PHP (Hypertext Preprocessor) là một ngôn ngữ lập trình mã nguồn mở được thực thi trên máy chủ.

## Câu hỏi 2: Tập tin PHP có phần mở rộng là gì ?

- Một tập tin PHP có phần mở rộng \*.php, nó có thể chứa các văn bản, mã nguồn HTML, CSS, JavaScript, Jquery, ... và mã PHP.

## Câu hỏi 3: PHP có gì khác so với HTML ?



yêu cầu xem trang web

gửi về nội dung trang web

yêu cầu xem trang web

gửi về nội dung trang web

phát sinh  
trang web



## Câu hỏi 4: Server làm gì đối với trang web PHP?

- Server chỉ quan tâm đến mã nguồn PHP, nó sẽ chuyển mã nguồn PHP sang HTML rồi gửi lại cho người dùng.
- Trang web PHP được server phát sinh chỉ được gửi đến một client duy nhất.
- Server PHP phải mạnh hơn nhiều lần so với một server HTML thông thường.

## Câu hỏi 5: Tại sao nên sử dụng PHP để lập trình web ?

- Chức năng: tạo ra những website động, thao tác với file trên server, nhận và gửi cookie, cập nhật database, hạn chế người dùng truy cập vào website, mã hóa dữ liệu, ...
- Ưu điểm: thực thi tốt trên các hệ điều hành, các máy chủ phổ biến hiện nay, kết hợp dễ dàng với các hệ quản trị cơ sở dữ liệu, tài liệu phong phú và đa dạng, cộng đồng sử dụng rộng rãi, được cung cấp hoàn toàn miễn phí, ...

## Câu hỏi 6: Có thể tạo một webserver ảo tại máy tính cá nhân ?



### XAMPP 1.8.1

*Một webserver ảo tại máy tính người dùng, đi kèm là các gói hỗ trợ như Apache, Php, Mysql, PhpMyAdmin, ...*

## Câu hỏi 7: Phần mềm gì để soạn thảo mã nguồn PHP?



**Zend Studio 9.0.3**

*IDE mạnh cho lập trình viên PHP*

**Lập trình PHP**  
[zend.vn](http://zend.vn)

# Phần 02 – Biến trong PHP

## Câu hỏi 1: Biến là gì ?

- Biến là một giá trị có thể thay đổi khi chương trình thực thi. Khi biến được tạo sẽ xuất hiện một vùng nhớ để lưu trữ

## Câu hỏi 2: Biến tồn tại bao lâu ?

- Biến trong PHP chỉ tồn tại trong thời gian server phát sinh trang web. Sau khi đã phát sinh xong trang web, tất cả các biến đều bị xóa đi.

## Câu hỏi 3: Làm sao để tạo biến trong PHP ?

- Một biến gồm 2 thành phần cơ bản: Tên biến và giá trị của biến

```
<?php  
    $firstName = "John";  
    $lastName = "Smith";  
    $number= 12;  
?>
```

## Câu hỏi 4: Khi đặt tên biến có cần theo quy định nào không ?

- Tên biến phải bắt đầu bằng một chữ cái hoặc ký tự gạch dưới (ký hiệu \_)
- Tên biến chỉ bao gồm các ký tự chữ, ký tự số và ký tự gạch dưới (Az, 09, \_)
- Không chứa ký tự khoảng trắng trong tên biến
- Phân biệt chữ hoa và chữ thường

## Câu hỏi 5: Tại sao có lúc đặt giá trị của biến trong dấu ngoặc kép có lúc lại không ?

- Khi các giá trị của biến được đặt trong dấu ngoặc kép (hoặc dấu ngoặc đơn) cho biết biến đó lưu trữ giá trị kiểu chuỗi
- Ngược lại cho biết biến đó lưu trữ giá trị kiểu số

## Câu hỏi 6: Ngoài kiểu chuỗi và kiểu số, trong PHP còn có kiểu dữ liệu nào nữa không ?

- Các kiểu dữ liệu thường được sử dụng trong PHP: String, Numeric, Boolean, Null, Array, Object
- Tạm thời chúng ta sẽ tập trung vào kiểu String và kiểu Numeric. Các kiểu dữ liệu khác chúng ta sẽ được giới thiệu sau.

## Câu hỏi 7: Làm sao biết được biến đó đang lưu giá trị thuộc kiểu dữ liệu nào ?

- Chúng ta có 2 cách sau để xác định kiểu dữ liệu của một biến nào đó
  - Sử dụng hàm `gettype()`
  - Sử dụng hàm `var_dump()`

## Câu hỏi 8: Có thể chuyển đổi kiểu dữ liệu của một biến nào đó hay không ?

- Chúng ta hoàn toàn có thể chuyển đổi kiểu dữ liệu của một biến nào đó, bằng cách thực hiện một trong hai cách sau:
  - Sử dụng cách ép kiểu
  - Sử dụng hàm **settype()**

## Câu hỏi 9: Có hàm nào để kiểm tra kiểu dữ liệu của một biến không ?

- Để kiểm tra kiểu dữ liệu của một biến nào đó chúng ta có thể dùng các hàm `is_numeric()`, `is_float()`, `is_string()`, `is_array()`, `is_object()`, ...

## Câu hỏi 10: Trong toán học ngoài biến số còn có khái niệm hằng số. PHP có hỗ trợ hằng số hay không ?

- Khác với biến, hằng số là giá trị không thể thay đổi được.
- Định nghĩa hằng

```
<?php  
    define("PI", 3.14);  
    echo "Value PI: " . PI;  
?>
```

# Phần 03 – Toán tử trong PHP

# Câu hỏi 1: Khái niệm toán tử ở đây bao gồm những nội dung gì ?

- Toán tử số học: + - \* / %
- Toán tử gán: += -= \*= /= %=
- Toán tử ++ --
- Toán tử so sánh > < >= <= == === != <> !==
- Toán tử logic AND OR XOR && || !
- Toán tử điều kiện

## Câu hỏi 2: Thực hiện + - \* / trong PHP như thế nào ?

```
<?php  
    $x = 2;  
    $y = $x + 2;
```

Các toán tử này nằm trong nhóm “Toán tử số học”, trong nhóm này chúng ta còn có thể thực hiện các phép toán

- % chia lấy phần dư
- – phủ định của một số

## Câu hỏi 3: Khi tôi viết \$x = \$x \* 5 có vẻ hơi dài dòng! Có cách viết ngắn gọn hơn không ?

```
<?php  
    $x = 2;  
    $x *= 5;
```

- Chúng ta có thể áp dụng cách viết này cho các trường hợp + - \* / %
- Cách viết này thuộc nhóm “Toán tử gán” : += -= /= \*= %=

## Câu hỏi 4: ++\$x và \$x++ có gì khác nhau

- `++$x` tăng `$x` lên một đơn vị, sau đó trả về giá trị của `$x`
- `$x++` trả về giá trị của `$x`, sau đó tăng `x` lên một đơn vị
- `--$x` giảm `$x` xuống một đơn vị, sau đó trả về giá trị của `$x`
- `$x--` trả về giá trị của `$x`, sau đó giảm `x` xuống một đơn vị

## Câu hỏi 5: Các phép so sánh > < >= <= PHP có hỗ trợ hay không ?

Các toán tử này nằm trong nhóm “Toán tử so sánh”, trong nhóm này chúng ta còn có thể thực hiện các phép toán

- == so sánh bằng
- === so sánh bằng tuyệt đối
- != so sánh khác
- !== so sánh khác tuyệt đối
- <> so sánh khác

## Câu hỏi 6: Toán tử logic là gì ?

| STT | Toán tử | Diễn giải   | Ví dụ                                      | Kết quả |
|-----|---------|---|--|---------|
| 1   | And     | \$x and \$y<br>Trả về true nếu x và y đều mang giá trị true   | \$x = 3; \$y = 6;<br>(\$x < 8 and \$y > 1) | true    |
| 2   | Or      | \$x or \$y<br>Trả về true nếu x hoặc y đều mang giá trị true  | (\$x >= 8 or \$y > 1)                      | true    |
| 3   | Xor     | \$x xor \$y<br>Trả về true nếu chỉ x hoặc y mang giá trị true | (\$x < 8 xor \$y > 1)                      | true    |
| 4   | &&      | \$x && \$y<br>Trả về true nếu x và y đều mang giá trị true    | (\$x > 8 && \$y > 1)                       | false   |
| 5   |         | \$x    \$y<br>Trả về true nếu x hoặc y đều mang giá trị true  | (\$x < 8 or \$y > 1)                       | true    |
| 6   | !       | !\$x<br>Trả về true nếu x false                               | ! (\$x===\$y)                              | true    |

## Câu hỏi 7: Toán tử điều kiện là gì ?

- Cú pháp (condition) ? value1 : value2;
- Ví dụ

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<?php
    $variable = "PHP training";
    $result = (is_string($variable)) ? "Chuỗi" : "Không phải chuỗi";
    echo $result;
```

# Phân 04 – Làm việc với Form

# Câu hỏi 1: Form dùng để làm gì ?

Đăng Nhập

Email :

Mật khẩu :

[Ghi nhớ tài khoản](#)

[Đăng nhập](#)

[Bạn không thể truy cập vào tài khoản ?](#)

Nhận các giá trị mà người dùng nhập vào và đưa ra các quá trình xử lý phù hợp.

## Câu hỏi 2: Tạo Form bằng cách nào ?

- Sử dụng mã nguồn HTML để tạo form

```
<form method="post" action="process.php" name="main-form">
    <input type="text" name="email" />
    <input type="text" name="password" />
</form>
```

- Method: cách thức dữ liệu được gửi đi (post hoặc get)
- Action: xác định trang / tập tin các dữ liệu được gửi đến để xử lý
- Name: tên của form

## Câu hỏi 3: Hai phương thức post và get có gì khác nhau ?

Sau khi người dùng tiến hành “submit form”, cả hai phương thức này đều tiến hành gửi dữ liệu đến server, tuy nhiên:

- **POST**

```
localhost/process.php
```

- **GET**

```
localhost/process.php?name=lan&age=25
```

## Câu hỏi 4: Làm sao để lấy các giá trị mà người dùng đã nhập trên FORM ?

- Đối với Form có method=“post” sử dụng **`$_POST`**
- Đối với Form có method=“get” sử dụng **`$_GET`**
- Sử dụng **`$_REQUEST`** cho cả 2 trường hợp trên

# Phần 05 – Phát biểu điều kiện

## Câu hỏi 1: Câu điều kiện là gì ?

- Câu điều kiện là câu lệnh mà chúng ta thường xuyên sử dụng khi viết mã cho bất kỳ ngôn ngữ lập trình nào.
- Câu điều kiện giúp chúng ta thực hiện những hành động khác nhau trong những điều kiện khác nhau.

## Câu hỏi 2: Có mấy loại câu điều kiện trong PHP ?

- Hai câu lệnh điều kiện thường được sử dụng trong PHP:
  - Câu điều kiện IF ... ELSE
  - Câu điều kiện SWITCH

## Câu hỏi 3: Sử dụng câu điều kiện IF như thế nào ?

- Trong nhóm điều kiện IF có 3 câu điều kiện: IF, IF ... ELSE, IF ... ELSE IF ... ELSE

```
<?php  
    $number = 12;  
    if($number >= 0) {  
        echo "Số dương";  
    }  
?>
```

## Câu hỏi 4: Cách sử dụng câu điều kiện SWITCH có gì khác so với câu điều kiện IF?

- Câu điều kiện Switch có một điều kiện mặc định, nghĩa là khi giá trị đưa vào không thỏa một điều kiện nào thì nó sẽ lấy các câu lệnh trong phần điều kiện mặc định để thực hiện

## Bài tập câu điều kiện SWITCH

### Mô phỏng máy tính điện tử

Số thứ nhất

25

Phép toán

%

Số thứ hai

2

Xem kết quả

Kết quả:  $25 \% 2 = 1$

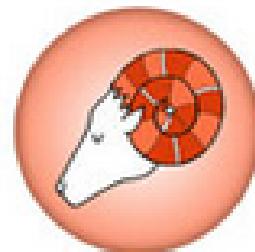
## Bài tập câu điều kiện

### Bạn thuộc chòm sao gì?

Ngày sinh

Tháng sinh

[Lấy chòm sao!](#)



Cung Kim Ngưu (*Taurus* - 21/4 - 21/5)

# Phần 06 – Vòng lặp

## Câu hỏi 1: Vòng lặp được hiểu như thế nào ?

- Vòng lặp là một đoạn mã lệnh trong chương trình được thực hiện lặp đi lặp lại cho đến khi thỏa mãn một điều kiện nào đó.

## Câu hỏi 2: Có bao nhiêu vòng lặp trong PHP ?

- Các vòng lặp thường sử dụng trong PHP:
  - For
  - While
  - Do ... While
  - For ... each

## Vòng lặp FOR

### Chép phạt

1. Em hứa sẽ làm bài tập ở nhà đầy đủ
2. Em hứa sẽ làm bài tập ở nhà đầy đủ
3. Em hứa sẽ làm bài tập ở nhà đầy đủ
4. Em hứa sẽ làm bài tập ở nhà đầy đủ
5. Em hứa sẽ làm bài tập ở nhà đầy đủ
6. Em hứa sẽ làm bài tập ở nhà đầy đủ

## Vòng lặp WHILE

### In tam giác

```
*  
**  
***  
****  
*****  
*****
```

```
*****  
*****  
****  
***  
**  
*
```

```
*  
***  
*****  
*****  
*****  
*****
```

```
*  
***  
*****  
*****  
*****  
*****
```

# Vòng lặp DO ... WHILE

Image gallery

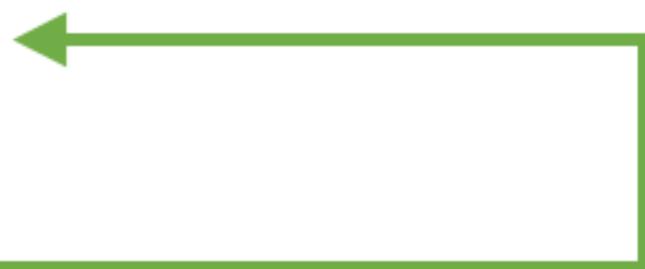


Show All

Lập trình PHP  
zend.vn

## BREAK - CONTINUE

```
while (condition) {  
    continue;  
    break;  
}
```



## BREAK - CONTINUE

- Câu lệnh **break** có chức năng thoát khỏi một vòng lệnh. Nó có thể được sử dụng để nhảy ra khỏi một vòng lặp.
- Câu lệnh **continue** có chức năng dừng vòng lặp tại giá trị đó và nhảy sang giá trị khác trong vòng lặp

## Bài tập 01

### Giải câu đố bằng vòng lặp

Yêu nhau cau sáu bồ ba

Ghét nhau cau sáu bồ ra làm mười

Mỗi người một miếng trăm người

Có mười bảy quả hỏi người ghét yêu.

*Hỏi có bao nhiêu người yêu nhau, ghét nhau?*

#### Đáp án

1. 10 người yêu nhau và 7 người ghét nhau

## Bài tập 02



### Mô phỏng máy ATM

Vui lòng nhập vào số tiền quý khách muốn thực hiện giao dịch

3450000

Rút tiền

#### Mệnh giá

Mệnh giá 500.000

Mệnh giá 200.000

Mệnh giá 50.000

#### Số lượng

6

2

1

#### Thành tiền

3.000.000

400.000

50.000

Tổng tiền: 3.450.000

# Phần 07 – Hàm trong PHP

## Câu hỏi 1: Hàm là gì ?

- Hàm là tập hợp một hay nhiều câu lệnh được xây dựng để thực hiện một chức năng nào đó.
- Khối lệnh này chỉ cần xây dựng duy nhất một lần, và có thể được sử dụng nhiều lần trong toàn bộ chương trình.

## Câu hỏi 2: Có bao nhiêu hàm trong PHP ?

- Hàm trong PHP được xây dựng vô cùng đa dạng và phong phú, bao gồm các hàm xử lý chuỗi, số, mảng, ngày tháng, ...
- Chúng ta tạm thời chia làm 2 nhóm hàm
  - Nhóm hàm được cung cấp sẵn bởi PHP
  - Nhóm hàm do người dùng tự định nghĩa

## Câu hỏi 3: Làm sao để viết một hàm trong PHP ?

- Vấn đề 01: Hàm không tham số và không có trả về
- Vấn đề 02: Sự trả về của hàm
- Vấn đề 03: Truyền tham số vào hàm
  - Phân biệt biến toàn cục và biến cục bộ
  - Phân biệt tham chiếu và tham trị
- Vấn đề 04: Tìm hiểu include và require

## Vấn đề 01: Hàm không tham số và không có trả về

- Xây dựng hàm vẽ các box
- Lưu ý về cách đặt tên hàm
- Biết cách khai báo và gọi hàm không có tham số và không có trả về

## Vấn đề 02: Sự trả về của hàm

- Xây dựng hàm vẽ các box với sự trả về của hàm return
- Hàm trả về một giá trị
- Hàm trả về nhiều giá trị
- Hàm trả về kết quả true hoặc false

## Vấn đề 03: Truyền tham số vào hàm

### Truyền tham số vào hàm

- Truyền nội dung vào cho box (hàm 1 tham số)
- Truyền chiều rộng và chiều cao của box (hàm nhiều tham số)
- Ghán giá trị mặc định cho tham số

## Vấn đề 03: Truyền tham số vào hàm

### Phân biệt biến toàn cục và biến cục bộ

- Local (biến cục bộ) là các biến được khai báo trong hàm và chỉ có thể được truy cập trong hàm đó. Biến cục bộ được xóa sau khi hàm của nó thực thi xong
- Global (biến toàn cục) là các biến được khai báo bên ngoài tất cả các hàm. Được sử dụng tại bất kỳ vị trí nào trong chương trình

## Vấn đề 03: Truyền tham số vào hàm

### Phân biệt tham chiếu và tham trị

- Khi truyền biến vào hàm theo kiểu tham trị. Sau khi kết thúc hàm giá trị của biến truyền vào không thay đổi
- Khi truyền biến vào hàm theo kiểu tham chiếu. Sau khi kết thúc hàm giá trị của biến truyền vào sẽ thay đổi tùy theo phần xử lý của hàm đó

## Vấn đề 04: Tìm hiểu include và require

- Câu lệnh include và require cùng có chức năng là kéo một file nào đó vào file hiện tại.  
Trong quá trình kéo file vào file hiện tại, nếu gặp lỗi:
  - Câu lệnh require sẽ ngừng thực hiện chương trình
  - Câu lệnh include sẽ tiếp tục thực hiện chương trình
- Tìm hiểu thêm câu lệnh include\_once và require\_once

## Chương 02

# Làm việc với ASNT

# Phân 01 – Giới thiệu

## Vấn đề 1: ASNT là gì ?

- ASNT từ viết tắt của Number, String, Array và Time. Đây là 4 kiểu dữ liệu mà chúng ta rất thường thao tác trong các ngôn ngữ lập trình.
- Trong chương học này, chúng ta sẽ được đặt ra các tình huống và xử lý các tình huống này khi thao tác với 4 kiểu dữ liệu trên.

## Vấn đề 2: Làm sao biết chúng ta đang thao tác với kiểu dữ liệu nào ?

- Sử dụng hàm var\_dump hàm gettype để lấy kiểu dữ liệu của một biến.
- Sử dụng nhóm hàm is\_numeric, is\_string, is\_array, is\_date để kiểm tra kiểu dữ liệu của một biến.

# Phân 01 – PHP Array

## Vấn đề 1: Hiểu như thế nào về array trong PHP ?

- Mảng là một biến đặc biệt và có thể lưu trữ nhiều giá trị.
- Một biến thông thường chỉ chứa một giá trị duy nhất, nếu chúng ta muốn chứa nhiều giá trị trong một biến thì biến đó phải là một mảng (Ví dụ cần lưu trữ thông tin của 1000 nhân viên)
- Trong PHP có 3 loại mảng: mảng số nguyên, mảng kết hợp và mảng đa chiều.

## Vấn đề 2: Khai báo và sử dụng mảng số nguyên

- Mảng số nguyên là mảng mà các chỉ số của các phần tử phải thuộc kiểu số nguyên (mảng số nguyên còn được gọi là mảng liên tục)
- Tìm hiểu cách truy cập vào phần tử của mảng và in mảng

## Vấn đề 3: Khai báo và sử dụng mảng kết hợp

- Mảng kết hợp là mảng mà các chỉ số của các phần tử có thể là chuỗi hoặc số  
(Mảng kết hợp còn gọi là mảng không liên tục)
- In danh sách các phần tử trong mảng kết hợp: foreach

## Vấn đề 4: Khai báo và sử dụng mảng đa chiều

- Mảng đa chiều là mảng mà mỗi phần tử trong mảng chính có thể là một mảng và mỗi phần tử trong mảng con lại cũng có thể là một mảng (Mảng đa chiều còn gọi là mảng lồng)
- In phần tử, in sách các phần tử trong mảng đa chiều: foreach

## Vấn đề 5: Lấy danh sách các khóa và danh sách các giá trị của một mảng nào đó ?

- array\_values (\$array) trả về một mảng liên tục có các phần tử có giá trị là giá trị lấy từ các phần tử của mảng \$array.
- array\_keys (\$array): trả về một mảng liên tục có các phần tử có giá trị là khóa lấy từ các phần tử của mảng \$array.

## Vấn đề 6: Loại bỏ phần tử ở đầu và cuối mảng ?

- array\_pop (\$array) loại bỏ phần tử cuối cùng của mảng. Hàm trả về phần tử cuối cùng đã được loại bỏ.
- array\_shift (\$array) loại bỏ phần tử đầu tiên của mảng. Hàm trả về phần tử đầu tiên đã được loại bỏ

## Vấn đề 7: Loại bỏ phần tử trùng nhau trong mảng?

- `array_unique ($array)` loại bỏ những phần tử trùng nhau trong mảng và trả về mảng mới

## Vấn đề 8: Xóa phần tử ở vị trí bất kỳ của mảng

- Sử dụng hàm unset để xóa bỏ phần tử ở vị trí bất kỳ trong mảng

## Vấn đề 9: Thêm một hoặc nhiều phần tử ở đầu hoặc cuối mảng ?

- `array_push ($array, $val1, $val2, ... , $valn)` thêm một hoặc nhiều phần tử vào cuối mảng `$array`. Hàm trả về kiểu số nguyên là số lượng phần tử của mảng `$array` mới
- `array_unshift ($array, $val1, $val2, ... , $valn)` thêm một hoặc nhiều phần tử vào đầu mảng `$array`. Hàm trả về kiểu số nguyên là số lượng phần tử của mảng `$array` mới

## Vấn đề 10: Đảo ngược vị trí các phần tử của mảng?

- `array_reverse ($array)` đảo ngược vị trí các phần tử của mảng, phần tử cuối trở thành phần tử đầu tiên, phần tử kế cuối trở thành phần tử thứ nhì, ... kết quả trả về là một mảng mới

## Vấn đề 11: Hoán đổi chỉ số và giá trị của mảng (đảo \$key và \$value) ?

- Sử dụng hàm array\_flip(\$array) trả về một mảng có khóa và giá trị được hoán đổi cho nhau so với mảng \$array (giá trị thành khóa và khóa thành giá trị)

## Vấn đề 12: Xác định tổng, giá trị lớn nhất và giá trị nhỏ nhất trong mảng ?

- Tính tổng các phần tử trong mảng array\_sum (\$array)
- Xác định phần tử nhỏ nhất trong mảng min(\$array)
- Xác định phần tử lớn nhất trong mảng max(\$array)

## Vấn đề 13: Thống kê số lần xuất hiện của các phần tử trong mảng ?

- Để thống kê sự xuất hiện của các phần tử trong mảng chúng ta sử dụng hàm array\_count\_values(array)

## Vấn đề 14: Kết hợp các mảng lại với nhau ?

- array\_merge (\$array1, \$array2, ..., \$arrayn) nhập 2 hay nhiều mảng thành một mảng duy nhất và trả về mảng mới

## Vấn đề 15: Lấy ngẫu nhiên chỉ số (\$key) của một mảng nào đó ?

- array\_rand (\$array, \$number) Lấy ngẫu nhiên \$number phần tử từ mảng \$array và đưa vào mảng mới (lấy giá trị khóa)

## Vấn đề 16: Tìm kiếm phần tử trong mảng

- array\_search (\$value,\$array) tìm phần tử mang giá trị \$value trong mảng \$array. Trả về khóa của phần tử tìm được

## Vấn đề 17: Kiểm tra một \$key hoặc \$value nào đó có tồn tại trong mảng hay không ?

- `array_key_exists ($key, $array)` kiểm tra khóa \$key có tồn tại trong mảng \$array hay không? Nếu có trả về giá trị true.
- `in_array ($value, $array)` kiểm tra giá trị \$value có tồn tại trong mảng \$array hay không? Nếu có trả về giá trị true.

## Vấn đề 18: Chuyển đổi các key trong mảng thành chữ hoa hoặc chữ thường

- Sử dụng hàm `array_change_key_case($array, case)` để chuyển đổi các chỉ số (`$key`) trong mảng thành chữ hoa hoặc chữ thường, tùy thuộc vào tham số `case` truyền vào. Kết quả trả về của hàm sẽ là một mảng mới

## Vấn đề 19: Chuyển đổi qua lại giữa mảng và chuỗi ?

- implode (\$str, \$array) chuyển các giá trị của mảng \$array thành một chuỗi bao gồm các phần tử cách nhau bởi ký tự \$str
- explode (\$delimiter, \$str) chuyển một chuỗi thành một mảng. Tách chuỗi dựa vào \$delimiter, mỗi đoạn tách ra sẽ thành một phần tử của mảng mới

## Vấn đề 20: Truy xuất phần tử của mảng với end, current, next và previous

- current(\$array) truy xuất phần tử hiện tại của mảng
- end(\$array) truy xuất phần tử cuối cùng của mảng
- next(\$array) truy xuất phần tử sau phần tử hiện tại của mảng
- prev(\$array) truy xuất phần tử trước phần tử hiện tại của mảng
- reset() quay về vị trí phần tử đầu tiên trong mảng

## Vấn đề 21: Chuyển đổi mảng về một chuỗi đặc biệt và ngược lại ?

- serialize (\$value) chuyển chuỗi/mảng/đối tượng \$value thành một chuỗi đặc biệt để lưu vào cơ sở dữ liệu
- unserialize (\$value) chuyển chuỗi đặc biệt được tạo từ serialize(\$value) về trạng thái ban đầu

## Vấn đề 22: Xáo trộn thứ tự các phần tử trong mảng ?

- Sử dụng hàm shuffle để tạo ra mảng mới (mảng liên tục) với thứ tự các phần tử trong mảng bị thay đổi

## Vấn đề 23: Tạo mảng từ các biến có sẵn ?

- Sử dụng hàm compact() để tạo ra mảng mới từ các biến có sẵn

## Vấn đề 24: Tạo mảng sử dụng hàm range()

- Sử dụng hàm range để tạo ra các phần tử của mảng

## Vấn đề 25: Tạo mảng bằng cách sử dụng hàm array\_combine?

- Sử dụng hàm `array_combine($key, $value)` để tạo một mảng mới có khóa được lấy từ mảng `$key` và giá trị được lấy từ mảng `$value` theo tuần tự

## Vấn đề 26: Các trường hợp so sánh giữa hai mảng ?

- **Trường hợp 1 : So sánh khác nhau**

- array\_diff (\$array1, \$array2) trả về một mảng bao gồm các phần tử có giá trị tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2
- array\_diff\_key (\$array1, \$array2) trả về một mảng bao gồm các phần tử có khóa tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2
- array\_diff\_assoc (\$array1, \$array2) trả về một mảng bao gồm các phần tử có khóa và giá trị tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2

## Vấn đề 26: Các trường hợp so sánh giữa hai mảng ?

- **Trường hợp 2 : So sánh giống nhau**

- array\_intersect (\$array1, \$array2) trả về một mảng bao gồm các phần tử giống nhau về giá trị giữa 2 mảng \$array1 và \$array2
- array\_intersect\_key (\$array1, \$array2) trả về một mảng bao gồm các phần tử giống nhau về khóa giữa 2 mảng \$array1 và \$array2
- array\_intersect\_assoc (\$array1, \$array2) trả về một mảng bao gồm các phần tử giống nhau về khóa và giá trị giữa 2 mảng \$array1 và \$array2

## Vấn đề 27: Xử lý giá trị các phần tử của mảng ?

- Hàm array\_walk sẽ gửi các giá trị của mảng đến một hàm nào đó để xử lý và nhận kết quả trả về là một mảng mới

## Vấn đề 28: Tìm hiểu hàm array\_map ?

- Hàm array\_map sẽ gửi các giá trị của một hay nhiều mảng đến một hàm nào đó để xử lý và nhận kết quả trả về là một mảng mới

## Vấn đề 29: Trích xuất một đoạn phần tử của mảng ?

- array\_slice (array, offset ,length, preserve) trích xuất lấy một đoạn phần tử của mảng từ từ vị trí bắt đầu offset (vị trí bắt đầu trong mảng là 0) và lấy length phần tử.

## Vấn đề 30: Thay thế một đoạn phần tử của mảng ?

- `array_splice(array1, offset ,length, array2)` xóa bỏ một đoạn phần tử của mảng array1 từ từ vị trí offset và lấy length phần tử. Sau đó thay thế các phần tử bị loại bỏ bằng mảng array2

## Vấn đề 31: Các trường hợp sắp xếp mảng

- **Sắp xếp theo giá trị**
  - sort(array) sắp xếp các phần tử trong mảng array tăng dần theo giá trị
  - rsort(array) sắp xếp các phần tử trong mảng array giảm dần theo giá trị
- **Sắp xếp theo khóa**
  - ksort(array) sắp xếp các phần tử trong mảng array tăng dần theo khóa
  - krsort(array) sắp xếp các phần tử trong mảng array giảm dần theo khóa

# Phần 02: Xử lý chuỗi

## Vấn đề 01: Khái niệm string được hiểu như thế nào?

- Khái niệm string được hiểu như là chuỗi, văn bản.
- Biến kiểu string được sử dụng để lưu trữ các giá trị có chứa ký tự. Các giá trị này luôn nằm trong cặp dấu nháy đôi hoặc dấu nháy đơn

## Vấn đề 02: Hiển thị ký tự nháy đơn và nháy đôi trong chuỗi ?

- Vì dấu nháy đơn và nháy đôi là các ký tự đặc biệt do đó để hiển thị các ký tự này trong chuỗi chúng ta sử dụng thêm ký tự \

## Vấn đề 03: Nối 2 hay nhiều chuỗi lại với nhau ?

- Sử dụng dấu chấm (kí hiệu .) để nối các giá trị kiểu chuỗi lại với nhau thành một giá trị duy nhất
- Lưu ý phân biệt giữa dấu cộng (+) và dấu chấm (.)
- Viết hàm nối 2 chuỗi bất kỳ bởi một ký tự nào đó

## Vấn đề 04: Đếm tổng số ký tự có trong chuỗi ?

- Tổng số ký tự có trong chuỗi, chúng ta thường gọi ngắn gọn là chiều dài của chuỗi.
- Trong PHP để lấy chiều dài của chuỗi, chúng ta sử dụng hàm strlen()
- Lưu ý trường hợp chuỗi có chứa các ký tự UTF-8 chúng ta sử dụng hàm mb\_strlen()

## Vấn đề 05: Đếm số từ có trong chuỗi ?

- Chúng ta sử dụng hàm str\_word\_count để đếm số từ xuất hiện trong chuỗi

## Vấn đề 06: Chuyển đổi chữ thường thành chữ hoa và ngược lại ?

- Để chuyển đổi chữ thường thành chữ HOA, chúng ta dùng hàm strtoupper(\$str).  
Ngược lại, ta dùng hàm strtolower(\$str)
- Chuyển đổi ký tự đầu tiên trong chuỗi thành chữ HOA ucfirst(\$str)
- Chuyển đổi ký tự đầu tiên trong chuỗi thành chữ thường lcfirst(\$str)
- Chuyển đổi tất cả các ký tự đầu tiên của các từ trong một chuỗi thành chữ in HOA ucwords(\$str)

## Vấn đề 07: Tìm kiếm vị trí xuất hiện của một từ nào đó trong chuỗi ?

- Sử dụng hàm strpos() để tìm kiếm chỉ số xuất hiện đầu tiên của một từ nào đó trong chuỗi
- Sử dụng hàm strrpos() để tìm kiếm chỉ số xuất hiện cuối cùng của một từ nào đó trong chuỗi
- Lưu ý strpos() và strrpost() dành cho PHP version 4

## Vấn đề 08: Đảo ngược một chuỗi

- Khi cần đảo ngược một chuỗi nào đó chúng ta sử dụng hàm strrev()

## Vấn đề 09: Trích xuất nội dung nào đó trong chuỗi

- Ví dụ lấy các ký tự từ vị trí thứ 2 đến vị trí thứ 6 trong một chuỗi nào đó. Khi gấp các yêu cầu dạng này chúng ta sử dụng hàm substr()

## Vấn đề 10: Xóa bỏ ký tự nằm bên trái chuỗi

- ltrim(\$str, \$params) sẽ xóa các ký nằm bên trái của một chuỗi nào đó.
- Tham số \$params khi bằng rỗng sẽ xóa bỏ các ký tự sau:
  - "\0" - NULL
  - "\t" - tab
  - "\n" - new line
  - "\x0B" - vertical tab
  - "\r" - carriage return
  - " " - ordinary white space

## Vấn đề 11: Xóa bỏ ký tự nằm bên phải chuỗi

- `rtrim($str, $params)` sẽ xóa các ký tự nằm bên phải của một chuỗi nào đó.
- Tham số `$params` khi bằng rỗng sẽ xóa bỏ các ký tự sau:
  - "\0" - NULL
  - "\t" - tab
  - "\n" - new line
  - "\x0B" - vertical tab
  - "\r" - carriage return
  - " " - ordinary white space

## Vấn đề 12: Xóa bỏ ký tự nằm bên trái và bên phải chuỗi

- `trim($str, $params)` sẽ xóa các ký tự nằm bên trái và bên phải của một chuỗi nào đó.
- Tham số `$params` khi bằng rỗng sẽ xóa bỏ các ký tự sau:
  - "\0" - NULL
  - "\t" - tab
  - "\n" - new line
  - "\x0B" - vertical tab
  - "\r" - carriage return
  - " " - ordinary white space

## Vấn đề 13: Kiểm tra chuỗi khác rỗng

- Sử dụng các hàm isset(), trim() để kiểm tra một chuỗi nào đó có khác rỗng hay không

## Vấn đề 14: Chuyển đổi qua lại giữa mảng và chuỗi ?

- implode (\$str, \$array) chuyển các giá trị của mảng \$array thành một chuỗi bao gồm các phần tử cách nhau bởi ký tự \$str
- explode (\$delimiter, \$str) chuyển một chuỗi thành một mảng. Tách chuỗi dựa vào \$delimiter, mỗi đoàn tách ra sẽ thành một phần tử của mảng mới

## Vấn đề 15: Trích xuất nội dung trong chuỗi

- Sử dụng hàm substr(\$str, \$start, \$length) để truy xuất các đoạn nội dung trong chuỗi
- Xây dựng hàm rút gọn chuỗi. Chúng ta thường gặp ở phần giới thiệu 1 bài viết, trang web chỉ xuất hiện 1 số từ được quy định sẵn và sau có thể là dấu ...

## Vấn đề 16: Lặp chuỗi

- Với yêu cầu lặp lại chuỗi \$str với số lần lặp là n, chúng ta sẽ gọi hàm str\_repeat(\$str, \$n)

## Vấn đề 17: Ký tự và mã ASCII

- `chr()` Trả về ký tự tương ứng với mã ASCII được truyền vào
- `ord()` Trả về giá trị ASCII của ký tự đầu tiên trong chuỗi

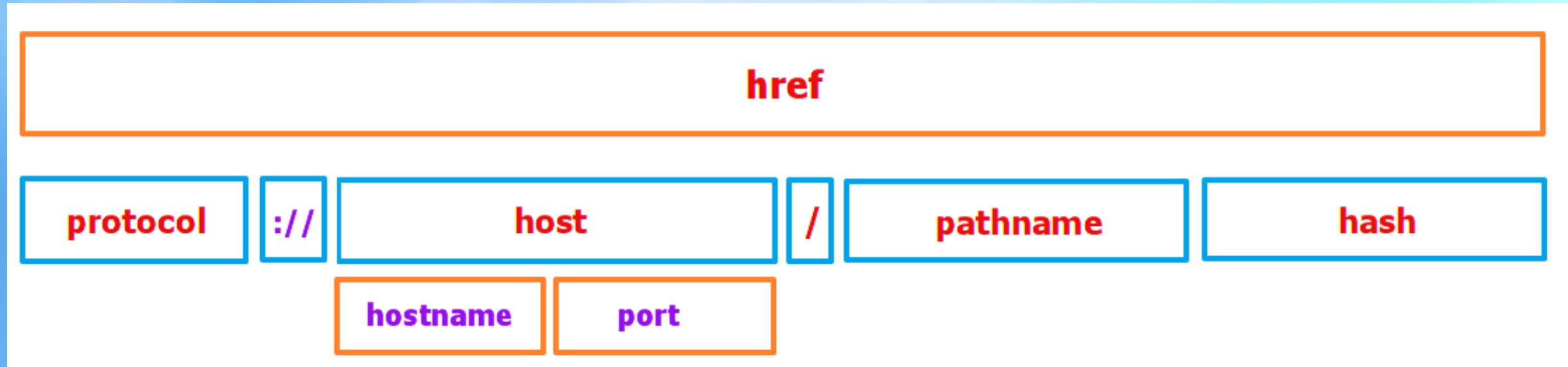
## Vấn đề 18: Phân tích chuỗi truy vấn

- Chúng ta sẽ sử dụng hàm `parse_str()` để chuyển các nội dung truy vấn vào các biến hoặc mảng

## Vấn đề 19: Phân tích URL

- Sử dụng hàm `parse_url` để truy xuất các thành phần protocol, domain name, path, .. của một URL nào đó

# Các thành phần cơ bản của URL



## Xác định các thành phần của URL

`http://www zend vn:8080/public/tin-cong-nghe/cd-o-viet-nam.html#title`

- Href: `http://www zend vn:8080/public/tin-cong-nghe/cd-o-viet-nam.html#title`
- Protocol: `http`
- Host: `www zend vn:8080`
- Hostname: `www zend vn`
- Port: `8080`
- Pathname: `public/tin-cong-nghe/cd-o-viet-nam.html#title`
- Hash: `#title`

# Bài tập 01: Truy xuất nội dung

Cho URL như sau:

[http://210.245.126.171/Music/NhacTre/TinhYeu\\_LyMaiTrang/wma32/06\\_BienTham\\_TinhYeu\\_LyMaiTrang.wma](http://210.245.126.171/Music/NhacTre/TinhYeu_LyMaiTrang/wma32/06_BienTham_TinhYeu_LyMaiTrang.wma)

Lấy các giá trị:

- No: 06
- Name: Bien Tham
- Album: Tinh Yeu
- Singer: Ly Mai Trang
- Type: wma

## Vấn đề 20: So sánh hai chuỗi

- `strcmp($str1, $str2)` so sánh hai chuỗi \$str1 và \$str2 với nhau
- `substr_compare ($str1, $str2, $start, $length)` lấy \$length phần tử từ vị trí \$start trong chuỗi \$str1 sau đó so sánh với chuỗi \$str2

## Vấn đề 21: Tăng độ dài của chuỗi với các ký tự mới

- Sử dụng hàm `str_pad($str, $length, $padString, $padType)` để tăng độ dài của chuỗi `$str` thành `$length` với các ký tự mới được thêm vào là `$pad_string` (cơ chế thêm là `$padType`)

## Vấn đề 22: Sắp xếp ngẫu nhiên thứ tự các ký tự trong chuỗi

- Sử dụng hàm str\_shuffle() sắp xếp ngẫu nhiên thứ tự các ký tự trong chuỗi

## Vấn đề 23: Tìm kiếm và thay thế ký tự

- Sử dụng hàm `str_replace($find, $replace, $string)` để thay thế giá trị `$find` trong chuỗi `$string` bằng giá trị `$replace`

## Vấn đề 24: Đếm số lần xuất hiện chuỗi con

- substr\_count(\$string, \$substring, \$start, \$length) lấy \$length phần tử từ vị trí \$start trong chuỗi \$str và thống kê số lần xuất hiện của \$substring trong chuỗi vừa lấy trên

## Vấn đề 25: Cắt chuỗi thành các phần tử của mảng

- Sử dụng hàm `str_split($str, $length)` cắt chuỗi thành từng phần tử trong mảng, mỗi phần tử có độ dài là `$length` ký tự

## Bài tập 02: Chuẩn hóa chuỗi

Một chuỗi được xem là đã được chuẩn hóa khi:

- Không có khoảng trắng ở đầu và cuối chuỗi
- Giữa các từ trong chuỗi chỉ tồn tại một khoảng trắng duy nhất
- Ký tự đầu tiên trong chuỗi phải là ký tự in hoa. Nếu chuỗi là một danh từ riêng yêu cầu các ký tự đầu tiên ở mỗi từ phải viết được viết hoa. Các ký tự còn lại ở dạng chữ thường.

## Vấn đề 26: Các trường hợp thao tác với ký tự gạch chéo \

- **addslashes(\$str)** thêm ký tự \ vào trước các ký tự ‘ “ \
- **addcslashes(\$str, \$character)** thêm ký tự \ vào trước ký tự \$character
- **stripslashes(\$str)** hiển thị chuỗi không có các ký tự gạch chéo được tạo bởi hàm addslashes
- **stripcslashes(\$str)** hiển thị chuỗi không có các ký tự gạch chéo được tạo bởi hàm addcslashes

## Vấn đề 27: Làm việc với các HTML entity

- **htmlspecialchars(\$str)** chuyển đổi các ký tự được quy định trước sang giá trị HTML entities
- **htmlspecialchars\_decode(\$str)** chuyển đổi các giá trị HTML entities được gọi bởi hàm htmlspecialchars () về giá trị ban đầu
- **htmlentities(\$str)** chuyển đổi các ký tự sang giá trị HTML entities
- **html\_entity\_decode(\$str)** chuyển đổi các giá trị HTML entities được gọi bởi hàm htmlentities(\$str) về giá trị ban đầu
- **get\_html\_translation\_table()** xem danh sách các giá trị HTML entities

## Vấn đề 28: Loại bỏ các thẻ HTML

- Sử dụng hàm `strip_tags($str)` để loại bỏ các thẻ HTML có trong chuỗi

## Bài tập 03: Đọc số có 3 chữ số

Viết chương trình đọc số có 3 chữ số, ví dụ

- 976 = Chín trăm bảy mươi sáu
- 206 = Hai trăm linh sáu
- 115 = Một trăm mười lăm
- 291 = Hai trăm chín mươi một

## Bài tập 04: Đọc số có 12 chữ số

Viết chương trình đọc số có 12 chữ số, ví dụ

- 123.456.789.123 = Một trăm hai mươi ba tỷ bốn trăm năm mươi sáu triệu bảy trăm  
tám mươi chín nghìn một trăm hai mươi ba đồng

# Phần 03: Làm việc với Number

## Vấn đề 01: Number trong PHP được hiểu như thế nào?

- Khái niệm Number được hiểu như là số, số trong PHP chúng ta quan tâm 2 loại: số nguyên và số thập phân

## Vấn đề 02: Làm cách nào kiểm tra biến có lưu giá trị kiểu Number hay không ?

- Sử dụng hàm `is_numeric` để kiểm tra biến có lưu trữ dữ liệu kiểu Number hay không ?
- Ngoài ra để kiểm tra:
  - Biến lưu trữ giá trị thuộc kiểu số nguyên `is_int`
  - Biến lưu trữ giá trị thuộc kiểu số thập phân `is_float`

## Vấn đề 03: Tạo dãy số

- Tạo ra một dãy số với số bắt đầu và số kết thúc được cho trước, chúng ta sử dụng hàm range(\$start, \$length, \$loop)

## Vấn đề 04: Làm tròn một số thập phân ?

- round() làm tròn đến số nguyên gần nhất
- ceil() làm tròn đến số nguyên gần nhất và lớn nhất
- floor() làm tròn đến số nguyên gần nhất và nhỏ nhất

## Vấn đề 05: Tìm tổng, số lớn nhất và nhỏ nhất của một dãy số

- Sử dụng hàm sort để thực hiện yêu cầu trên
- Sử dụng phương thức có sẵn min, max, array\_sum()

## Vấn đề 06: Làm sao tạo ra một số ngẫu nhiên trong PHP ?

- Sử dụng hàm rand(min, max) giá trị ngẫu nhiên được trả về nằm trong đoạn [min,max]
- Ví dụ hiển thị hình ảnh ngẫu nhiên trong tập hợp hình ảnh cho trước

## Vấn đề 07: Định dạng số

- Sử dụng hàm number\_format để định dạng cách hiển thị giữa các phần nghìn trong 1 số

## Vấn đề 08: Lấy giá trị tuyệt đối

- Sử dụng hàm `abs($number)` để trả về giá trị tuyệt đối của một số nào đó

## Vấn đề 09: Lũy thừa

- Hàm pow(x, y) trả về kết quả là  $x^y$

## Vấn đề 10: Căn bậc 2

- Sử dụng hàm sqrt(\$number) để tính căn bậc hai của \$number

## Vấn đề 11: Các hàm tính toán số học

- Các hàm tính toán logarithm: `log()`, `log10()`, `log1p()`
- Các hàm tính toán lượng giác: `sin()`, `cos()`, `tan()`, `rad2deg()`, `pi()`
- Tài liệu tham khảo [http://www.w3schools.com/php/php\\_ref\\_math.asp](http://www.w3schools.com/php/php_ref_math.asp)

## Bài tập phân số

- Viết chương trình tối giản một phân số nào đó
- Thực hiện phép cộng hai phân số
- Thực hiện phép trừ hai phân số
- Thực hiện phép nhân hai phân số
- Thực hiện phép chia hai phân số

# Phần 04: Thao tác với thời gian

## Vấn đề 01: Lấy thời gian hiện tại ở máy chủ

- Sử dụng hàm **getdate()** để lấy thời gian hiện tại được thiết lập ở máy chủ

## Vấn đề 02: Làm việc với múi giờ

- Trả về kết quả múi giờ đã được thiết lập sẵn: **date\_default\_timezone\_get()**
- Thiết lập múi giờ: **date\_default\_timezone\_set()**
- Xem danh sách các múi giờ: **timezone\_identifiers\_list()**

## Vấn đề 03: Xác định khoảng thời gian hiện tại so với timestamp (01/01/1970)

- Unix Timestamp là dạng thời gian thường được dùng trên hệ thống Unix. Là số giây được tính từ thời điểm 01/01/1970 lúc 0 giờ 0 phút 0 giây theo giờ GMT
- `time()` trả về số giây từ thời điểm hiện tại so với 01/01/1970
- `mktime()` trả về số giây tại một thời điểm nào đó so với 01/01/1970

## Vấn đề 04: Định dạng cách hiển thị thời gian với date()

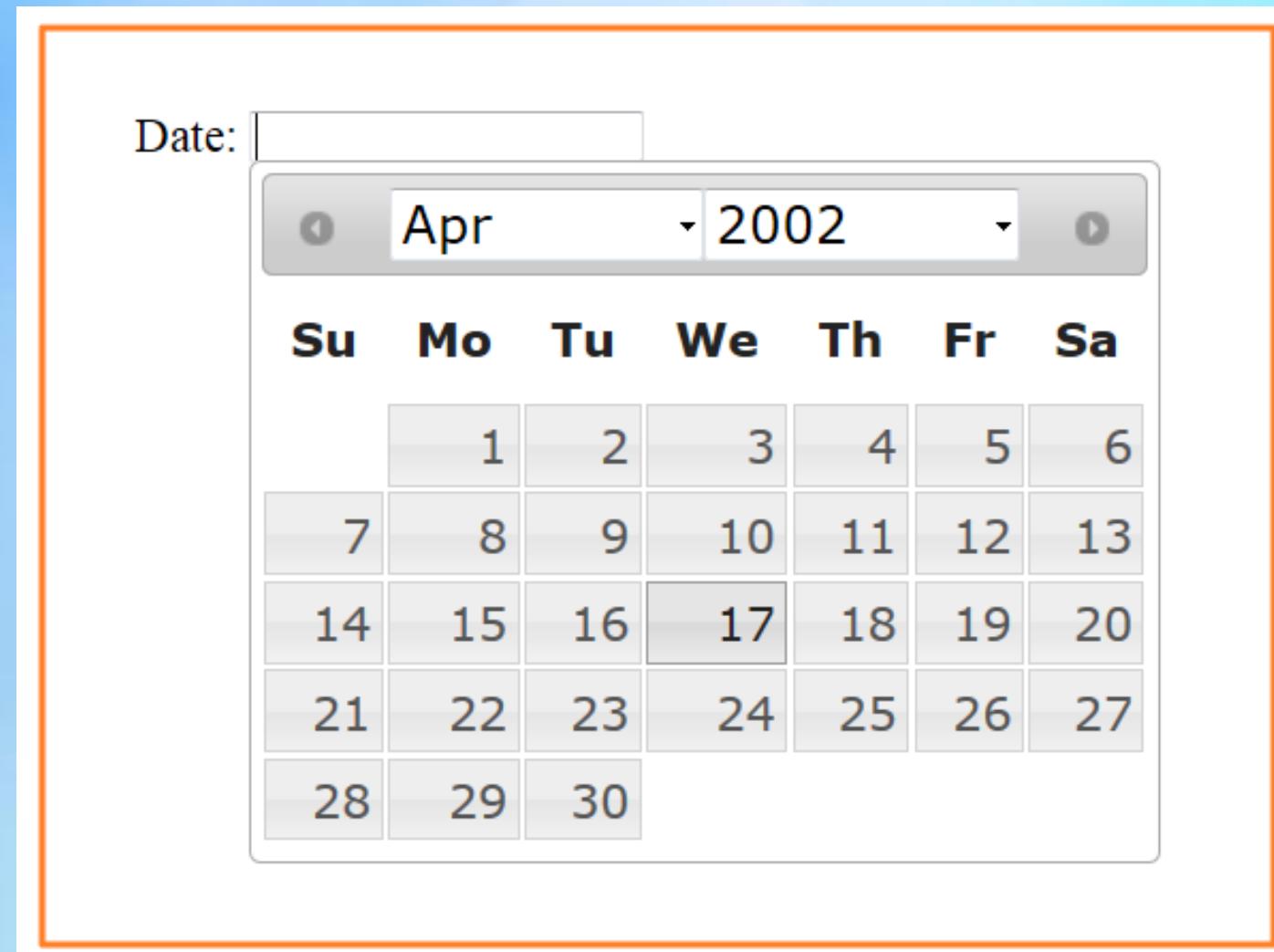
| Ký hiệu | Đại diện | Giá trị       |
|---------|----------|---------------|
| d       | Day      | 01 → 31       |
| J       | Day      | 1 → 31        |
| m       | Month    | 01 → 12       |
| n       | Month    | 1 → 12        |
| M       | Month    | Jan, Feb, ... |
| Y       | Year     | 2013          |
| y       | Year     | 13            |

| Ký hiệu | Đại diện | Giá trị |
|---------|----------|---------|
| g       | Hour     | 1 → 12  |
| G       | Hour     | 0 → 23  |
| h       | Hour     | 01 → 12 |
| H       | Hour     | 01 → 23 |
| i       | Minute   | 0 → 59  |
| s       | Second   | 00 → 59 |
| a       | Am / pm  | am pm   |
| A       | Am / pm  | AM PM   |

## Vấn đề 05: Kiểm tra ngày hợp lệ

- **checkdate(month,day,year)** kiểm tra các giá trị truyền vào có tạo thành một ngày hợp lệ hay không?

## Vấn đề 06: Sử dụng jQuery UI - Datepicker



## Bài tập

1. Định dạng lại cách hiển thị ngày khi sử dụng Datepicker
2. Hiển thị thời gian dạng: 01:57 PM Thứ 3, ngày 18/06/2013
3. Kiểm tra năm nhuận
4. Tìm số ngày trong tháng của một năm nào đó
5. Xác định khoảng thời gian giữa hai ngày (như ở ứng dụng Facebook )

## Bài tập

6. So sánh hai ngày bất kỳ
7. Xây dựng hàm để thực hiện thao tác cộng, trừ thời gian

# Chương 03:

# Các thành phần khác trong PHP

# Phần 01: PHP Regex

# Ôn tập Regular Expression

## Ví dụ 01: Kiểm tra email hợp lệ !

Kiểm tra giá trị email hợp lệ:

1. Địa chỉ email phải bắt đầu bằng một ký tự.
2. Địa chỉ email là tập hợp của các ký tự a-z, 0 đến 9 và có thể có các ký tự như dấu chấm ( . ), dấu gạch dưới ( \_ )
3. Độ dài tối thiểu của email là 5 ký tự và độ dài tối đa là 32 ký tự
4. Tên miền của email có thể là tên miền cấp 1 hoặc tên miền cấp 2

→ Pattern: ^[a-zA-Z][a-zA-Z0-9\_\.]{4,31}@([a-zA-Z0-9]{2,})(\.[a-zA-Z0-9]{2,4}){1,2}\$

## Ví dụ 02: Kiểm tra tên đăng nhập hợp lệ !

Kiểm tra giá trị tên đăng nhập hợp lệ:

1. Tên đăng nhập phải bắt đầu bằng một ký tự hoặc dấu gạch dưới
2. Tên đăng nhập là tập hợp của các ký tự a-z, 0-9 và có thể có các ký tự như dấu chấm ( . ), dấu gạch dưới ( \_ ), khoảng trắng
3. Độ dài tối thiểu của tên đăng nhập là 5 ký tự và độ dài tối đa là 32 ký tự

→ Pattern: ^[a-zA-Z\_][a-zA-Z0-9\_\.\\s]{4,31}\$

## Ví dụ 03: Kiểm tra mật khẩu hợp lệ

Kiểm tra giá trị mật khẩu hợp lệ:

1. Mật khẩu là tập hợp của các ký tự a-z, 0-9 có thể có các ký tự như dấu chấm ( . ), dấu gạch dưới ( \_ ) và các ký tự đặc biệt
2. Mật khẩu phải có chiều dài 8 ký tự
3. Phải tồn tại ít nhất 1 ký tự đặc biệt, 1 ký tự in hoa và 1 chữ số nào đó

→ Pattern: ^(?=.\*\d)(?=.\*[A-Z])(?=.\*\W).{8,8}\$

## Ví dụ 04: Kiểm tra địa chỉ website hợp lệ

Các địa chỉ website sau là các địa chỉ hợp lệ

1. http://www zend vn
2. https://www zend vn vn
3. http://zend vn
4. https://zend vn
5. www zend vn

Hãy sử dụng biểu thức chính quy để mô tả được các địa chỉ trên !

→ Pattern: ^([https://\(www\.\)?|\(www\.\)](https://(www\.)?|(www\.)))[\[a-z0-9\-\-\]{3,}](#)([\.\[a-z\]{2,4}](#))[{1,2}](#)\$

# Tìm hiểu hàm preg\_match và preg\_match\_all

# Sử dụng PHP Regex kiểm tra giá trị đầu vào

1. Địa chỉ email hợp lệ
2. Tên đăng nhập hợp lệ
3. Mật khẩu hợp lệ
4. Địa chỉ website hợp lệ

# Tìm hiểu hàm preg\_replace

# Kỹ thuật quét bảng ngoại hối của Vietcombank

| Tỷ giá | Lãi suất     | Biểu phí         |           |
|--------|--------------|------------------|-----------|
| Mã NT  | Mua tiền mặt | Mua chuyển khoản | Bán       |
| AUD    | 19,072.70    | 19,187.83        | 19,492.31 |
| EUR    | 27,112.13    | 27,193.71        | 27,625.23 |
| GBP    | 31,346.64    | 31,567.61        | 32,004.46 |
| JPY    | 207.30       | 209.39           | 212.71    |
| USD    | 21,230.00    | 21,230.00        | 21,246.00 |

(Áp dụng tại Hội sở chính NHTMCP Ngoại thương Việt Nam)

Xem thông tin tỷ giá tại các chi nhánh [tại đây](#)

 Để nhận thông tin cập nhật tỷ giá,  
đăng ký tại đây

Lập trình PHP  
zend.vn

# Kỹ thuật quét tin tức của VNExpress



## Container văng 2 bánh, lao vào cây xăng

11:05

Sáng nay, chiếc container đang chạy trên quốc lộ 22, khi tới đoạn qua xã Tân Thới Nhì (huyện Hóc Môn, TP HCM) bất ngờ đâm nát dải phân cách, húc gãy trụ điện và đâm đầu vào trạm xăng.



## Hủ tục rùng rợn giữa thăm sơn cùng cốc

09:01

Lối sống hồn nhiên, bần năng hết minh cộng với tư duy đơn giản của đồng bào Tây Nguyên xa xưa đã sinh ra hủ tục hà khắc trong những buôn làng biệt lập giữa thăm sơn cùng cốc.

- Phép thuật kỳ bí của tộc người Rục



## Gần 30 trường công bố điểm thi

11:54

Sáng 24/7, HV Chính sách Phát triển, ĐH Hồng Đức và ĐH Tài chính Ngân hàng Hà Nội công bố điểm thi. Hiện 28 trường đã có kết quả.

- HV Âm nhạc Huế, ĐH Tiền Giang, Thủ Dầu Một công bố điểm

# Kỹ thuật quét tin tức của Dantri



## Liên tục bị từ chối, MU “đánh cú chót” với Fabregas

(Dân trí) - Chỉ trong vòng 1 tuần, các nhà ĐKVĐ Premier League đã liên tục bị Barcelona “ngó lơ” 2 lời đề nghị hỏi mua Fabregas. Đường cùng, MU đã quyết định “đánh cú chót” bằng việc ra giá 35 triệu bảng cho tiền vệ ngôi sao này.

Xem tiếp ▶



## Logo các đội bóng Premier League: Sự thay đổi và những điều đặc biệt (P1)

(Dân trí) - Mỗi đội bóng đều có một huy hiệu riêng mang tính biểu tượng của đội bóng. Thiết kế logo các đội bóng có thể thay đổi dần qua năm tháng nhưng luôn mang theo rất nhiều điều đặc biệt về đội bóng mà nhiều người chưa biết đến.

Xem tiếp ▶

Lập trình PHP  
zend.vn

# Ôn tập

1. Ôn tập lại các ký hiệu thường được sử dụng của RE
2. Tìm hiểu các hàm preg\_match, preg\_match\_all và preg\_replace
3. Áp dụng RE trong PHP để xử lý các trường hợp
  - Kiểm tra các giá trị đầu vào
  - Tìm kiếm và thay thế chuỗi
  - Quét tin tức

# Phần 02: PHP File

## Vấn đề 01: Kiểm tra sự tồn tại của tập tin, thư mục

Sử dụng hàm `file_exists($fileName)` để kiểm tra sự tồn tại của tập tin, thư mục

- `$filename`: tên (đường dẫn) tập tin, thư mục cần kiểm tra
- Kết quả trả về `true` → Tồn tại
- Kết quả trả về `false` → Không tồn tại

## Vấn đề 02: Xem một số thông tin cơ bản của tập tin, thư mục

- `filetype($fileName)`  trả về kiểu của \$filename (tập tin hoặc thư mục)
- `filesize($fileName)`  trả về dung lượng của \$filename (đơn vị bytes)
- `is_readable($fileName)`  kiểm tra \$fileName có được quyền đọc hay không ?
- `is_writeable($fileName)`  kiểm tra \$fileName có được quyền ghi hay không ?
- `is_executable($fileName)`  kiểm tra \$fileName có được quyền thực thi hay không ?

## Vấn đề 03: Các hàm lấy thông tin từ đường dẫn

- **basename(\$path)** trả về kết quả là tên của tập tin từ đường dẫn \$path
- **dirname(\$path)** trả về tên thư mục tại đường dẫn \$path
- **pathinfo (\$path, \$options)** trả về một mảng các thông tin từ đường dẫn \$path
  - dirname
  - basename
  - extension

## Vấn đề 04: Thống kê số dòng, số từ và số ký tự trong file

- `file ($fileName)` đọc tập tin \$fileName thành một mảng, mỗi dòng trong tập tin \$fileName tương ứng với một phần tử của mảng
- `file_get_contents ($fileName)` đọc tập tin \$fileName thành một chuỗi

## Vấn đề 05: Ghi nội dung vào tập tin với file\_put\_contents

- Sử dụng hàm **file\_put\_contents** (\$fileName, \$data, \$mod) để ghi nội dung \$data vào tập tin \$fileName
- Quá trình thực thi của hàm **file\_put\_contents()**: Tạo ra tập tin nếu tập tin đó chưa tồn tại → Mở tập tin → Ghi nội dung → Đóng tập tin
- Nếu **\$mod = FILE\_APPEND** nội dung cũ ở tập tin được giữ, nội dung mới được ghi vào cuối tập tin
- Nếu ghi thành công, trả về tổng số ký tự đã ghi ngược lại trả về FALSE

## Vấn đề 06: Đổi tên tập tin, thư mục

- Sử dụng hàm `rename ($oldName, $newName)` để đổi tên tập tin, thư mục từ giá trị `$oldName` thành giá trị `$newName` (lưu ý thay đổi cả phần đường dẫn)
- Hàm trả về kết quả TRUE nếu rename thành công, trả về kết quả FALSE nếu rename không thành công

## Vấn đề 07: Sao chép tập tin

- Sử dụng hàm copy (\$sourceFile, \$destinationFile) để copy nội dung từ tập tin \$sourceFile sang tập tin \$destinationFile
- Hàm trả về kết quả TRUE nếu copy thành công, trả về kết quả FALSE nếu copy không thành công
- Làm sao copy một thư mục nào đó ?

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P1)

Xét mối quan hệ giữa người dùng và một tập tin, thư mục. Chúng ta có 3 nhóm sau đây:

- Owner/User: Người sở hữu, chủ tài khoản.
- Group: Các tài khoản cùng hoạt động trong một nhóm.
- Other/Guest: Ngoài 2 đối tượng trên.

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P2)

Xét về quyền thao tác giữa người dùng và một tập tin. Chúng ta có 3 loại quyền sau:

- Read: có thể đọc nội dung trong tập tin
- Write: có thể thay đổi nội dung của tập tin
- Execute: có thể mở tập tin (khởi chạy chương trình nếu là một tập tin ứng dụng)

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P3)

Xét về quyền thao tác giữa người dùng và **một thư mục**. Chúng ta có 3 loại quyền sau:

- Read: Có thể duyệt nội dung trong thư mục (Xem trong thư mục có gì, phải đi kèm với quyền Execute).
- Write: Có thể thay đổi nội dung trong thư mục này (Thêm, xóa, sửa, đổi tên thư mục/tập tin bên trong thư mục này).
- Execute: Có thể mở thư mục (Mở thư mục ra, nhưng không thể thấy gì trong thư mục này nếu không đi kèm với quyền Read).

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P4)

Cấp quyền truy cập cho tập tin, thư mục:

- Quyền cho một tập tin, thư mục thường được biểu diễn tương ứng với một số nguyên có 3 chữ số xyz
  - Số thứ nhất x - đại diện cho quyền hạn của Owner/User.
  - Số thứ hai y - đại diện cho quyền hạn của Group.
  - Số thứ ba z - đại diện cho quyền hạn của Other/Guest.

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P5)

Giá trị của các chữ số xyz, quy định như sau:

- Read = 4
- Write = 2
- Execute = 1

→ Read + Write + Execute =  $4 + 2 + 1 = 7$  (Toàn quyền)

→ Read + Execute =  $4 + 1 = 7$  (Chỉ có thể đọc và thực thi tập tin)

## Vấn đề 08: Khái quát về vấn đề phân quyền tập tin, thư mục (P6)

Một tập tin được cấu hình với quyền 754, như vậy chúng ta có các thông tin sau:

- Chữ số đầu tiên  $7 = 4 + 2 + 1 \rightarrow$  quyền: toàn quyền  $\rightarrow$  Owner/User có toàn quyền
- Chữ số thứ hai  $5 = 4 + 1 \rightarrow$  quyền: đọc và thực thi  $\rightarrow$  Group có quyền đọc và thực thi
- Chữ số thứ ba  $4 \rightarrow$  quyền: đọc  $\rightarrow$  Other/Guest chỉ có quyền đọc tập tin.

## Vấn đề 09: Tạo – Xóa – Cấp quyền cho thư mục

- Sử dụng hàm **mkdir (\$path, \$mode)** để tạo ra thư mục ở đường dẫn \$path, với quyền truy cập vào thư mục là \$mode
- Sử dụng hàm **rmdir (\$path)** để xóa thư mục ở đường dẫn \$path
- Sử dụng hàm **fileperms (\$dirName)** để xem quyền truy cập đối với thư mục \$dirName
- Sử dụng hàm **chmod (\$dirName, \$mod)** để cấp quyền truy cập cho thư mục \$dirName

## Vấn đề 10: Tạo – Xóa – Cấp quyền cho tập tin

- Sử dụng hàm `file_put_contents ($path, null)` để tạo ra tập tin ở đường dẫn `$path`
- Sử dụng hàm `unlink ($path)` để xóa tập tin ở đường dẫn `$path`
- Sử dụng hàm `fileperms ($fileName)` để xem quyền truy cập đối với tập tin `$fileName`
- Sử dụng hàm `chmod ($fileName, $mod)` để cấp quyền truy cập cho tập tin `$fileName`

## Vấn đề 11: Liệt kê danh sách tập tin và thư mục

- Sử dụng hàm `glob ($pattern)` để lấy danh sách các tập tin và thư mục với tên thỏa `$pattern`
- Sử dụng tham số thứ hai `GLOB_ONLYDIR` để kết quả trả về chỉ bao gồm các thư mục.

## Vấn đề 12: Tìm hiểu hàm parse\_ini\_file (1)

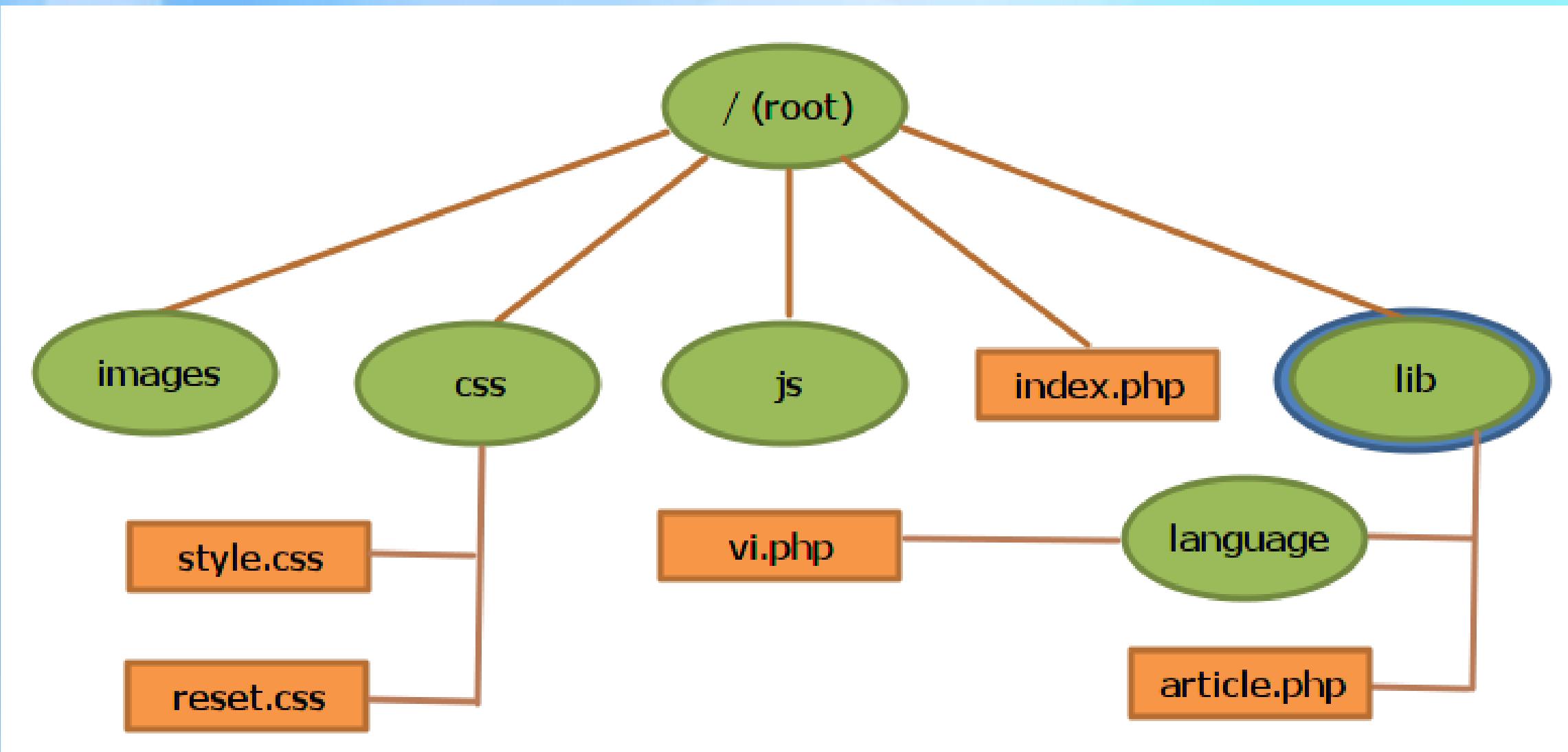
- Cấu trúc tập tin \*.ini

```
1
2 ; comment
3
4 [info]
5 name=Nguyễn Văn A
6 age=23
7
8 [study]
9 school=Nguyễn Đình Chiểu
10 programming=PHP
11 |
```

## Vấn đề 12: Tìm hiểu hàm parse\_ini\_file (2)

- Sử dụng hàm `parse_ini_file ($fileName)` để đọc nội dung \$fileName thành một mảng.
- Lưu ý nếu muốn đọc luôn các phần section cần truyền giá trị TRUE vào tham số thứ hai

## Vấn đề 13: Một số định nghĩa về hệ thống cây thư mục (1)



## Vấn đề 13: Một số định nghĩa về hệ thống cây thư mục (2)

- **Root-directory:** thư mục cao nhất trong cây thư mục và trên nó không còn có một thư mục nào khác. Root-directory được đánh dấu với ký hiệu /
- **Working-directory (current directory):** thư mục mà người sử dụng đang làm việc
- **Parent-directory:** thư mục cha - thư mục nằm ngay phía trên một thư mục khác trong cây thư mục
- **Path (đường dẫn):** mỗi tệp tin hay thư mục trong hệ thống linux được chỉ định bởi một đường dẫn (để truy cập vào tập tin hay thư mục đó)

## Vấn đề 13: Phân biệt đường dẫn tuyệt đối – Đường dẫn tương đối

- Đường dẫn tuyệt đối:
  - Là những đường dẫn được tính từ root (/), đi qua các thư mục khác cho đến tập tin hoặc thư mục đích
  - Khi chúng ta đang ở article.php, đường dẫn tuyệt đối đến các tập tin
    - style.css sẽ là /style/style.css
    - vi.php sẽ là /lib/language/vi.phpstyle.css
  - Thường được dùng với các hàm include, require để nhúng các tập tin .php vào một tập tin php nào đó hoặc dùng để khai báo thư mục chứa hình ảnh trên host

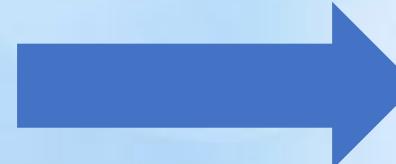
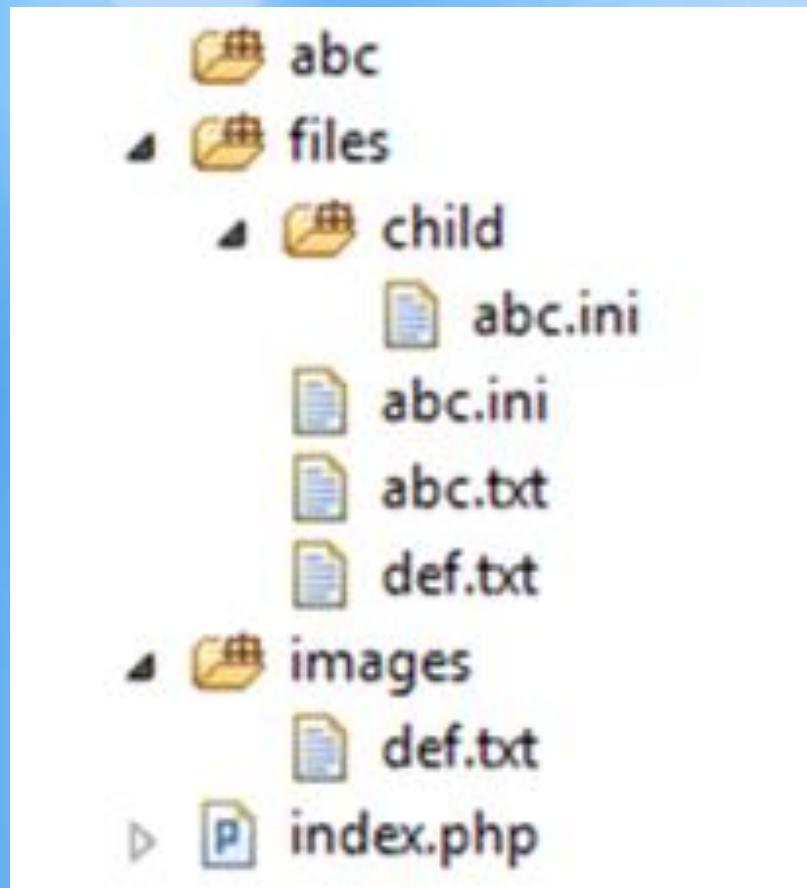
# Vấn đề 13: Phân biệt đường dẫn tuyệt đối – Đường dẫn tương đối

- Đường dẫn tương đối:
  - Là những đường dẫn có điểm xuất phát là thư mục hiện tại (current directory).
  - Khi chúng ta đang ở article.php, đường dẫn tương đối đến các tập tin
    - style.css sẽ là ../css/style.css
    - vi.php sẽ là /lib/language/vi.php
  - Thường được sử dụng để load các tập tin javascript, css và hình ảnh lên và hiển thị trong trang web

## Vấn đề 14: Thao tác với thư mục

- `getcwd()` trả về thư mục hiện tại
- `realpath()` trả về đường dẫn tuyệt đối của đường dẫn \$path
- `chdir()` thay đổi thư mục hiện tại
- `dir()` mở một thư mục lên và thao tác với thư mục đó (3 thao tác read, rewind, close )
- `opendir()` mở một thư mục lên và thao tác với thư mục đó
- `closedir()` đóng thư mục được mở bởi hàm opendir()
- `scandir()` lấy danh sách các tập tin và thư mục của đường dẫn \$path

## Vấn đề 15: Liệt kê cấu trúc tập tin thư mục đến cấp 2



- D: abc
- D: files
  - F: abc.ini
  - F: abc.txt
  - D: child
    - F: def.txt
- D: images
  - F: def.txt
- F: index.php

## Bài tập: Một số yêu cầu bổ sung

- Khi danh sách rỗng: hiển thị thông báo, ẩn button Delete File (**index.php**)
- Thay đổi thông báo trong trường hợp Add và Edit File (**add.php, edit.php**)
- Truyền ID của file chưa tồn tại: hiển thị thông báo (**edit.php, delete.php**)
- Xóa nhiều file: Hiển thị một hộp thoại xác nhận lại yêu cầu xem người dùng có chắc chắn muốn xóa các file đó hay không ? (**multy-delete.php**)

# Phần 03: PHP Recursive

## Vấn đề 01: Tìm hiểu Hàm đệ quy

- Hàm đệ quy là hàm có sự gọi lại chính nó. Trong hàm đệ quy đòi hỏi phải có điểm dừng.
- Hình thức đơn giản của một hàm đệ quy

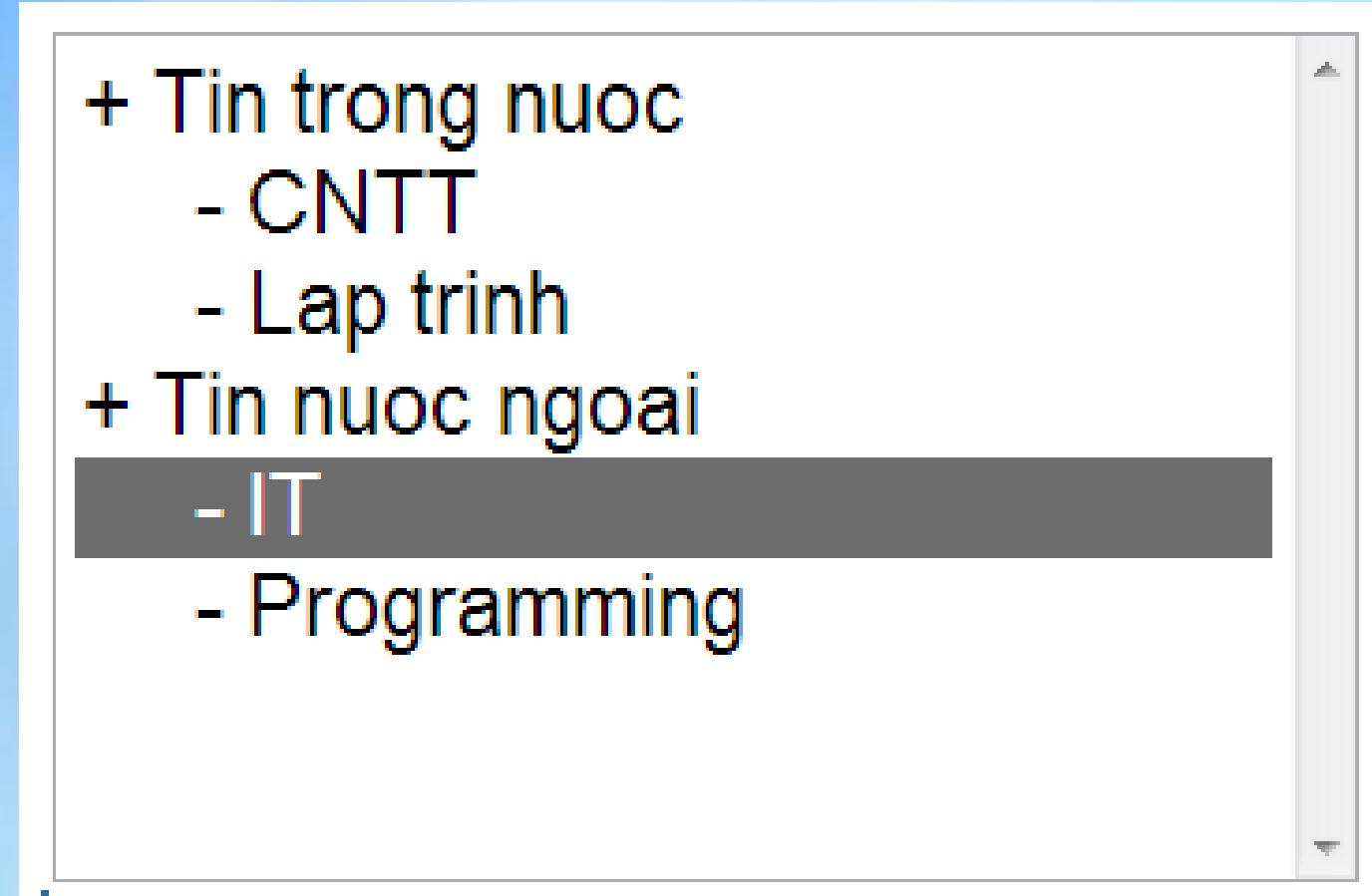
```
function myFunction($param1, $param2){  
    if($param1 == $param2){ // điều kiện dừng hàm  
        // Trả về giá trị - Kết thúc công việc  
    }else{  
        // gọi lại hàm myFunction($param1, $param2)  
    }  
}
```

## Vấn đề 02: Xây dựng hàm đệ quy tính n!

## Vấn đề 03: Xây dựng hàm đệ quy tính tổng từ 1 đến n

## Vấn đề 04: Xây dựng hàm đệ quy liệt kê menu đa cấp (sử dụng thẻ DIV)

# Vấn đề 05: Xây dựng hàm đệ quy liệt kê menu đa cấp (sử dụng selectbox)

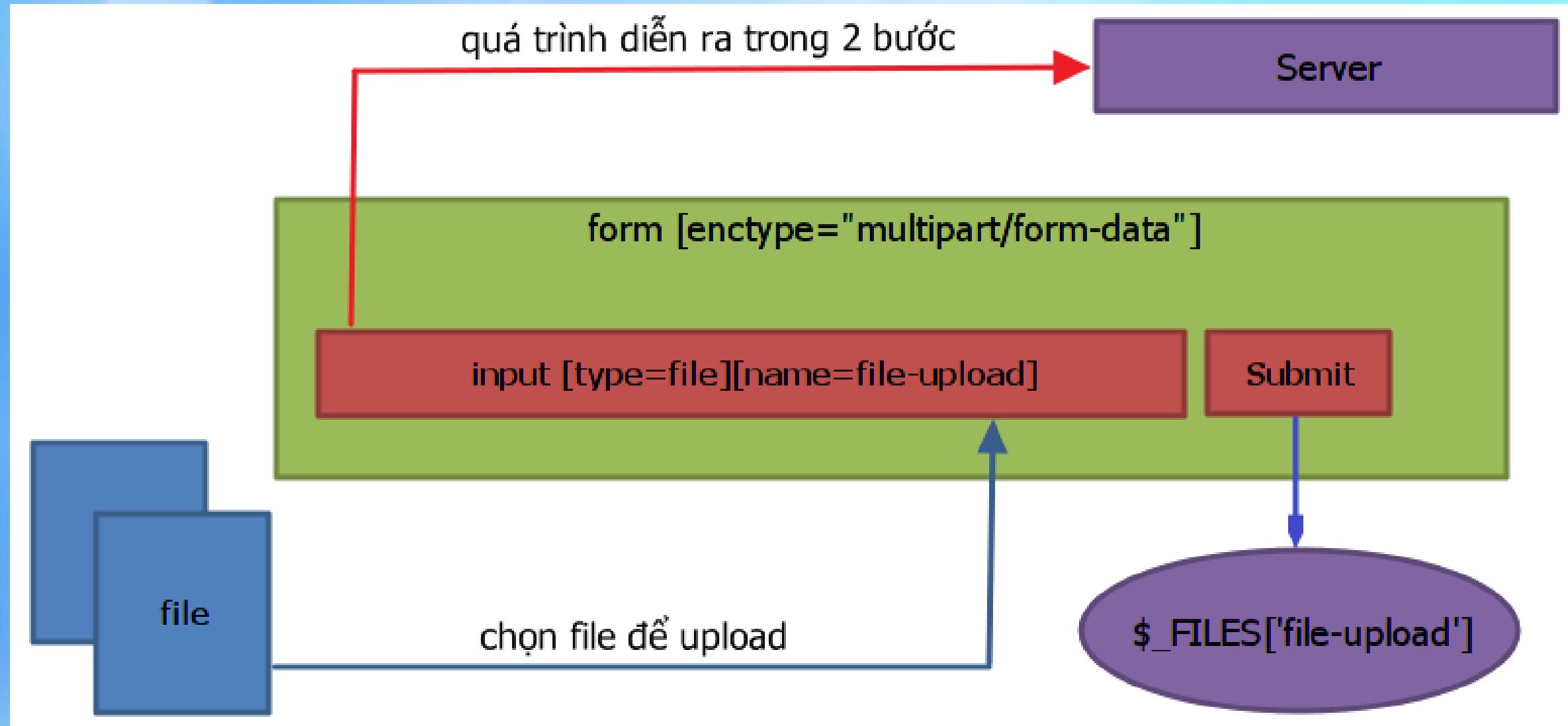


## Vấn đề 06: Xây dựng hàm đệ quy liệt kê menu đa cấp (sử dụng thẻ UL)

**Vấn đề 07: Xây dựng hàm đệ quy liệt kê tất cả tập tin và thư mục tại một đường dẫn nào đó**

# Phần 04: PHP File Upload

# Vấn đề 01: Cơ chế upload file trong PHP (1)



## Vấn đề 01: Cơ chế upload file trong PHP (2)

Giả sử tập tin upload có tên là picture.jpg, quá trình upload gồm 2 bước sau:

- Bước 01: Copy picture.jpg vào bộ nhớ tạm và đổi tên thành \*.tmp
- Bước 02:
  - Di chuyển file tạm lên server
  - Đổi tên file tạm thành picture.jpg

## Vấn đề 02: Xây dựng form HTML cho upload file

```
<form action="#" method="post" name="main-form" id="main-form" enctype="multipart/form-data">  
    <input type="file" name="file-upload" />  
    <input type="submit" name="submit-button" />  
</form>
```

## Vấn đề 03: Tìm hiểu các tham số của file upload

1. `$_FILE['file-upload']['name']`: Tên file (file-upload) upload lên server
2. `$_FILE['file-upload']['size']`: Kích thước của file
3. `$_FILE['file-upload']['type']`: Kiểu file
4. `$_FILE['file-upload']['tmp_name']`: Tên thư mục tạm trên server để chứa file
5. `$_FILE['file-upload']['error']`: Thông báo lỗi khi upload file

## Vấn đề 04: Upload file

Sử dụng hàm move\_upload\_file (\$filename , \$destination) để upload tập tin \$fileName vào vị trí \$destination

## Vấn đề 05: Xây dựng ứng dụng Upload file

Tạo một chương trình cho phép upload file với các yêu cầu sau đây:

1. Chỉ cho phép upload các file có kích thước tối thiểu là 100 Kb và kích thước tối đa là 5Mb
2. Chỉ cho phép upload các tập tin có phần mở rộng là jpg, png, zip, mp3

## Vấn đề 06: Xây dựng ứng dụng Upload multiple file

Tạo một chương trình cho phép upload file với các yêu cầu sau đây:

1. Chỉ cho phép upload các file có kích thước tối thiểu là 100 Kb và kích thước tối đa là 5Mb
2. Chỉ cho phép upload các tập tin có phần mở rộng là jpg, png, zip, mp3
3. Xây dựng tập tin \*.ini cấu hình cho các yêu cầu trên

# Phần 05: PHP Filter

## Vấn đề 01: PHP Filter là gì ?

- PHP Filter (phiên bản 5.2.0) **kiểm tra** nguồn dữ liệu trước khi thực hiện một vấn đề gì đó. Thuật ngữ này trong lập trình thường gọi là validate và filter
- Validate: dùng để xác nhận hoặc kiểm tra xem dữ liệu có đáp ứng đủ điều kiện yêu cầu hay không ?
- Filter: dùng để lọc các dữ liệu đưa vào không phù hợp với yêu cầu và trả về một kết quả đúng với dữ liệu yêu cầu.

## Vấn đề 02: Sử dụng Filter để kiểm tra dữ liệu

- Sử dụng hàm `filter_var($variable, $filter, $option)` để kiểm tra biến `$variable` với điều kiện đưa vào, biến `$filter` thường mang các giá trị
  - `FILTER_VALIDATE_BOOLEAN`
  - `FILTER_VALIDATE_EMAIL`
  - `FILTER_VALIDATE_FLOAT`
  - `FILTER_VALIDATE_IP`
  - `FILTER_VALIDATE_URL`
  - ...

## Vấn đề 03: Tự định nghĩa và gọi hàm trong PHP Filter

- Sử dụng hàm **filter\_var(\$variable, FILTER\_CALLBACK, array('options'=>'myFunc'))** để kiểm tra biến \$variable với hàm myFunc();

## Vấn đề 04: Sử dụng RE trong PHP Filter

- Sử dụng hàm `filter_var($variable, FILTER_VALIDATE_REGEXP, $option)` để kết hợp thêm RE trong filter
- Thực hiện các yêu cầu sau:
  - Kiểm tra số điện thoại với định dạng sau 084-08-38.212121
  - Kiểm tra phần mở rộng của FILE có phải là .jpg .gif .png
  - Kiểm tra giá trị nhập vào có phải là số và chữ hay không

## Vấn đề 05: Sử dụng hàm filter\_var\_array()

- Sử dụng hàm **filter\_var\_array()** để kiểm tra, lọc cùng lúc nhiều giá trị

## Vấn đề 06: Tìm hiểu hàm filter\_input()

- Hàm **filter\_input(input\_type, variable, filter, options)** được sử dụng để kiểm tra các giá trị từ FORM. Chức năng này cũng có thể lấy được dữ liệu từ các trường hợp khác như \$\_GET, \$\_POST, \$\_COOKIE, \$\_ENV, \$\_SERVER
  - input\_type: kiểu lấy dữ liệu (INPUT\_GET, INPUT\_POST, INPUT\_COOKIE, ... )
  - variable: tên của phần tử bạn muốn lấy dữ liệu
  - filter: kiểu filter
  - options: mảng tham số

## Vấn đề 07: Tìm hiểu hàm filter\_input\_array()

- Tương tự hàm **filter\_input** tuy nhiên hàm **filter\_input\_array()** cho phép thực hiện việc validate và filter cùng lúc trên nhiều giá trị khác nhau

# Phần 06: Error & Exception

## Vấn đề 01: Các lỗi cú pháp thường gặp trong lập trình PHP

- Đặt tên biến hoặc tên hàm không theo quy tắc, truyền thiếu tham số vào hàm.
- Lỗi chính tả: người viết mã có thể viết hay gọi sai tên hàm, tên biến.
- Vượt quá khả năng tính toán: Khi người lập trình yêu cầu chức năng quá khả năng của server sẽ gây ra các lỗi mà đôi khi không xác định được

→ Các lỗi cú pháp đa phần đều được nhắc nhở bởi trình soạn thảo mã nguồn (editor)

## Vấn đề 02: Sử dụng câu lệnh die()

- Câu lệnh die(\$variable) in ra nội dung của biến \$variable và kết thúc chương trình ngay tại vị trí gọi lệnh die()
- Câu lệnh die() được sử dụng khá thường xuyên trong việc debug

## Vấn đề 03: Ẩn lỗi đối với người dùng

- Không hiển thị các phần thông báo lỗi cho người dùng, gây ảnh hưởng không tốt đối với người dùng và website → ẩn lỗi bằng cách nào ?
- Ẩn các lỗi phát sinh khi website được vận hành, ghi các lỗi vào một file log nào đó để có thể xem lại và khắc phục các lỗi này.

```
12  
13     ini_set('display_errors', 'off');  
14     ini_set('log_errors', 'on');  
15     ini_set('error_log', 'php.error.log');  
16
```

## Vấn đề 04: Thiết lập các phần thông báo được hiển thị

Thiết lập loại thông báo lỗi muốn hiển thị error\_reporting() hoặc ini\_set('error\_reporting', VALUE)

- Ẩn tất cả các lỗi: error\_reporting(0);
- Hiển thị tất cả các lỗi: error\_reporting(E\_ALL);
- Hiển thị các phần warning: error\_reporting(E\_WARNING);
- Hiển thị các phần notice: error\_reporting(E\_NOTICE);
- Giá trị hiển thị được thiết lập mặc định: error\_reporting(E\_ALL ^ E\_NOTICE);

## Vấn đề 05: Creating a Custom Error Handler

Dựa vào `error_function(error_level, error_message, error_file, error_line, error_context)` để xây dựng lại nội dung thông báo muốn hiển thị

- `error_level`: error report level (`E_NOTICE`, `E_WARNING`, `E_ALL`, ...)
- `error_message`: nội dung thông báo lỗi muốn hiển thị
- `error_file`: tên file xảy ra lỗi
- `error_line`: xảy ra lỗi ở dòng nào

## Vấn đề 06: Exception

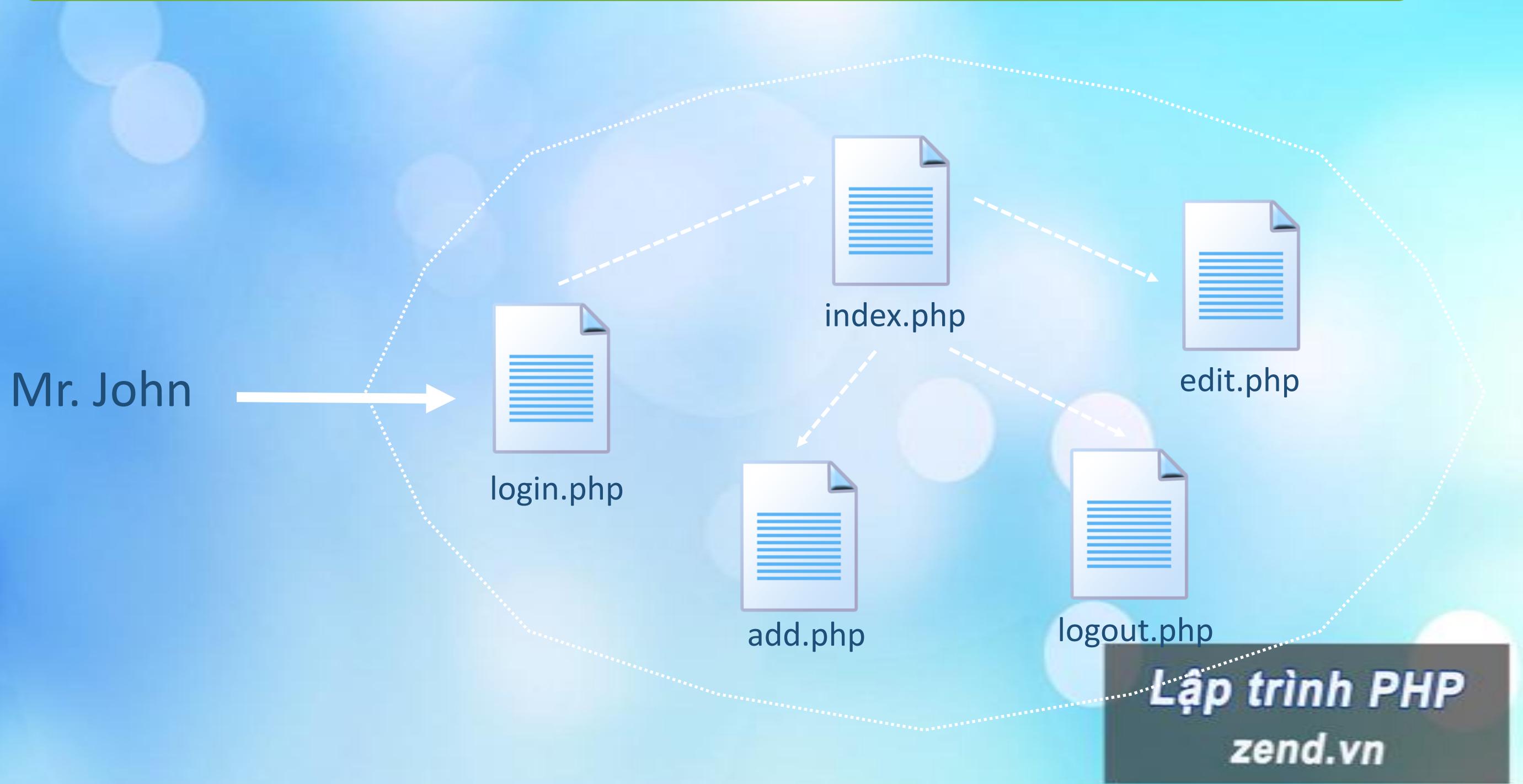
- [http://www.w3schools.com/php/php\\_exception.asp](http://www.w3schools.com/php/php_exception.asp)
- <http://php.net/manual/en/language.exceptions.php>
- [http://www.tutorialspoint.com/php/php\\_error\\_handling.htm](http://www.tutorialspoint.com/php/php_error_handling.htm)

# Phần 07: Session & Cookie

## Vấn đề 01: PHP Session

- PHP session cho phép lưu trữ dữ liệu của người dùng trên server để sử dụng ở các lần sau ( như username, thông tin đặt hàng ...).
- Các thông tin session này chỉ là tạm thời và thường bị xoá đi ngay khi người dùng rời khỏi trang web đã dùng session.
- Mỗi session sẽ được cấp một định danh (ID) khác nhau và nội dung được lưu trong thư mục thiết lập trong file php.ini (tham số session.save\_path).
- Session hoạt động bằng cách tạo 1 chuỗi unique (UID) cho từng vistor và chứa thông tin dựa trên ID đó.

## Vấn đề 02: Mô tả vấn đề (1)



## Vấn đề 02: Mô tả vấn đề (2)

- Mỗi trang PHP là một chương trình đang chạy trên máy chủ. Khi chương trình bắt đầu chạy, nó yêu cầu hệ điều hành của máy chủ cấp cho nó một số bộ nhớ để lưu trữ các biến.
- Khi trang được tải xong, hệ điều hành của máy chủ lấy bộ nhớ trả lại → bộ nhớ của trang sẽ bị xóa → tất cả thông tin trên trang đều bị xóa
- Dữ liệu được lưu vào bộ nhớ ở mỗi trang là độc lập với nhau (trang A không truy cập được bộ nhớ của các biến được tạo bởi trang B)



`$x → 1207  
$page → 3`

## Vấn đề 02: Mô tả vấn đề (3)



## Vấn đề 02: Mô tả vấn đề (4)

SERVER

John's  
session  
memory

Clara's  
session  
memory

Louise's  
session  
memory

Mary's  
session  
memory

Joe's  
session  
memory

Jim's  
session  
memory

## Vấn đề 03: Sử dụng session trong PHP

- `session_start()` khởi tạo session, lưu ý hàm này phải được đặt trước thẻ `<html>`
- Cập nhật thông tin session thông qua biến `$_SESSION`
- Xóa session
  - `unset()`
  - `session_destroy()`

## Vấn đề 04: Session có thể lưu trữ các giá trị gì ?

- Lưu biến vào session
- Lưu mảng vào session
- Lưu hàm (function) vào session
- Lưu file (nội dung tập tin) vào session
- Lưu hình ảnh vào session

## Vấn đề 05: session\_encode & session\_decode

- session\_encode() chuyển đổi các nội dung được lưu trong SESSION thành một chuỗi đặc biệt
- session\_decode() phân tích chuỗi đặc biệt được chuyển đổi bởi hàm session\_encode() và lưu vào SESSION

## Vấn đề 06: Tìm hiểu COOKIE

- PHP session và PHP cookie xét cho cùng đều được sử dụng để lưu trữ dữ liệu của người dùng.
- Đối với Session các dữ liệu này được lưu tại server, ngược lại đối với cookie các dữ liệu này được lưu trữ ngay tại trình duyệt web của người sử dụng
- Mỗi browser quản lý và lưu trữ cookie theo cách riêng của nó, cho nên 2 browser cùng truy cập vào 1 website sẽ nhận được 2 cookie khác nhau.

## Vấn đề 07: Thao tác với cookie

- **Tạo cookie** setcookie(\$name, \$value , \$time)
- **Truy cập cookie** \$\_COOKIE
- **Hủy cookie**
  - setcookie(\$name)
  - setcookie(\$name, \$value, time()-3600)

## Vấn đề 08: So sánh cookie và session

- Cookie và session đều được sử dụng để lưu trữ dữ liệu. Tuy nhiên session được lưu tại server, cookie được lưu ở client
- Session bị xóa khi người dùng đóng trình duyệt, cookie bị xóa khi thời gian tồn tại của nó kết thúc
- Session bảo mật hơn cookie, cookie được lưu ở client nên có thể được chỉnh sửa bởi client

# Phần 08: PHP Mail

# Vấn đề 01: Cấu hình XAMPP 1.8 để thực hiện việc gửi mail (1)

## Tập tin php.ini (D:\xampp\php\php.ini)

```
1 extension = php_openssl.dll  
2 extension = php_sockets.dll  
3 [mail function]  
4 ; SMTP = mail.gmail.com  
5 ; smtp_port = 25  
6 ; sendmail_from = postmaster@localhost  
7 sendmail_path = "\"D:\\xampp\\sendmail\\sendmail.exe\" -t"  
8 ; sendmail_path = "D:\\xampp\\miltodisk\\miltodisk.exe"  
9 ;mail.force_extra_parameters =  
10 mail.add_x_header = Off  
11 mail.log = "D:\\xampp\\php\\logs\\php_mail.log"  
12
```

# Vấn đề 01: Cấu hình XAMPP 1.8 để thực hiện việc gửi mail (1)

## Tập tin sendmail.ini (D:\xampp\sendmail\sendmail.ini)

```
1 [sendmail]
2
3 smtp_server = smtp.gmail.com
4 smtp_port = 465
5 smtp_ssl = ssl
6 default_domain = localhost
7 error_logfile = error.log
8 debug_logfile = debug.log
9 auth_username = lthlan54@gmail.com
10 auth_password = yourpassword
11 pop3_server =
12 pop3_username =
13 pop3_password =
14 force_sender =
15 force_recipient =
16 hostname = localhost
17
```

## Vấn đề 02: Tìm hiểu cách sử dụng hàm mail()

mail (\$to, \$subject, \$message,\$headers) thực hiện chức năng gửi mail, các tham số có ý nghĩa như sau

- **\$to** : địa chỉ người nhận email
- **\$subject** : tiêu đề của email
- **\$message** : nội dung email (có thể bổ sung các thẻ html)
- **\$header** : chèn thêm các header vào email. Các header này không nằm trong phần nội dung của email mà dùng để quản lý việc gửi email (ví dụ chèn thêm các trường CC, BCC khi gửi email)

# Chương 04

# Thao tác với XML

# Phần 01: Tìm hiểu XML

## Vấn đề 01: XML là gì ?

- Ngôn ngữ đánh dấu mở rộng (eXtensible Markup Language)
- XML không có thẻ riêng, người dùng có thể tạo bất kỳ thẻ nào theo ý muốn (tuân theo quy tắc của XML)
- Các thẻ XML khá giống với HTML: tag, data, attribute
- XML được xây dựng theo dạng cây, phải có tối thiểu một nút gốc
- XML được hỗ trợ trên các trình duyệt phổ biến hiện nay

## Vấn đề 02: XML dùng để làm gì ?

- Lưu trữ thông tin nhỏ
- Tạo phần tóm tắt nội dung cho website (RSS)
- Tạo sơ đồ cho website (sitemap)
- Là cầu nối trao đổi dữ liệu giữa các ứng dụng web (web service)
- ...

## Vấn đề 03: Tạo file XML

- Tạo file \*.xml lưu trữ thông tin một quyển sách gồm:
  - Tên quyển sách
  - Tác giả
  - Số trang
  - Cân nặng (đơn vị: gram, kilogram)
- Tạo file \*.xml lưu trữ thông tin nhiều quyển sách

## Vấn đề 04: Ghi chú trong XML

```
<?xml version="1.0" encoding="utf-8"?>

<!--
<book>
    <title>Lập trình jQuey</title>
    <img file="jquery.png" />
</book>
-->

<book>
    <title>Lập trình jQuey & Zend Framework</title>
    <img file="jquery.png" />
</book>
```

## Vấn đề 05: Các quy tắc viết tài liệu XML

- Mỗi tài liệu XML phải có một phần tử gốc và nó là duy nhất
- Phải có thẻ đóng khi mở thẻ
- Phải có thẻ đóng của phần tử con trước khi đóng phần tử cha
- Tên các phần tử của XML phân biệt hoa thường
- Giá trị thuộc tính đặt trong dấu nháy đôi
- Không sử dụng các giá trị đặc biệt & (&amp;) < (&lt;) > (&gt;) " ("quot;) ' (&apos;)

## Vấn đề 06: Lưu trữ các ký hiệu đặc biệt trong XML

- Trong trường hợp cần lưu trữ các ký hiệu đặc biệt trong file XML chúng ta sử dụng CDATA

```
<content>
  <![CDATA[
    <title>Lập trình jQuey</title>
    <author>ZendVN Group</author>
  ]]>
</content>
```

## Vấn đề 07: XSL là gì ?

- XSL là một ngôn ngữ chuẩn giúp chúng ta chuyển đổi tài liệu XML thành một tài liệu dễ đọc hơn đối với người dùng.
- Nhúng tập tin XSL vào tập tin XML
- Hiển thị nội dung XML
- Sử dụng câu điều kiện trong XSL
- Sử dụng vòng lặp trong XSL

## Vấn đề 08: Xpath – Các giá trị đặc biệt ở đường dẫn

- **nodeName** Lấy tất cả phần tử có tên là nodeName
- **Ký hiệu /** Lựa chọn từ phần tử gốc của tập tin XML
- **//nodeName** Truy cập phần tử có tên nodeName ở vị trí bất kỳ
- **Ký hiệu .** lấy toàn bộ giá trị của phần tử hiện tại
- **Ký hiệu ..** lấy toàn bộ giá trị của phần tử cha
- **Ký hiệu @** lấy giá trị của thuộc tính

## Vấn đề 09: Xpath – Truy cập phần tử

File XML của chúng ta có 4 phần tử book, chúng ta có các cách truy cập như sau

- **book[2]** Lấy ra thông tin phần tử book ở vị trí thứ 2
- **book[last()]** Lấy ra thông tin phần tử book ở vị trí cuối cùng
- **book[last()-1]** Lấy ra thông tin phần tử book ở vị trí kế cuối
- **book[@id]** Lấy ra thông tin phần tử book có thuộc tính id
- **book[@id=2]** Lấy ra thông tin phần tử book có thuộc tính id bằng 2

## Vấn đề 10: Xpath – Một số vấn đề khác

- Phép toán so sánh < > = !=
- Phép toán số học + - \* /
- Sử dụng hàm format\_number để định dạng lại số
- Sử dụng hàm round để làm tròn số

# Phần 02: SimpleXML

## Vấn đề 01: Tạo tập tin XML đơn giản

## Vấn đề 02: Sử dụng SimpleXML chuyển chuỗi thành object (XML)

## Vấn đề 03: Sử dụng SimpleXML chuyển object thành chuỗi (XML)

## Vấn đề 04: Sử dụng SimpleXML truy xuất các phần tử của tập tin XML

## Phần 03: Tìm hiểu DOMDocument

## Vấn đề 01: Xây dựng tập tin XML

- Tạo file XML lưu thông tin quyển sách
- Tạo node book
- Tạo các node con (author, title, pages, ...) của node boook
- Tạo các thuộc tính cho các node
- Chuyển array chứa thông tin quyển sách sang XML

## Vấn đề 02: Truy xuất tập tin xml

- Đọc tập tin XML
- Xác định tên node hiện tại
- Xác định node cha của node hiện tại
- Xác định thuộc tính của node
- Truy xuất vào các node con

## Vấn đề 03: Thêm nút , Xóa nút và Thay thế nút

Giả sử chúng ta có \$parentNode, \$pagesNode và \$weightNode là các đối tượng node. Trong đó \$parentNode là cha của \$pageNode

- `$parentNode->insertBefore ($weightNode, $pagesNode)` thêm node weight vào node parent, và node weight nằm trước node page
- `$parentNode->removeChild ($pagesNode)` xóa node pages ra khỏi node parent
- `$parentNode->replaceChild ($weightNode, $pagesNode)` thay node pages bằng node weight và là con của node parent

## Vấn đề 04: Sử dụng XPATH với DOMXpath

# Chương 05

# Lập trình hướng đối tượng

# Phần 01: Class & Object

# Lập trình hướng hướng thủ tục & hướng đối tượng

## Lập trình hướng thủ tục

- Giải quyết vấn đề từng bước đến khi đạt yêu cầu
- Lập trình từ trên xuống,
- Lập trình theo hàm → chỉ tạo ra hàm xử lý khi gặp vấn đề nào đó

## Lập trình hướng đối tượng

- Dựa trên nền tảng các lớp đã xây dựng sẵn
- Xác định trước các chức năng cần phải thực hiện

# Lập trình hướng đối tượng

- OOP – Object oriented programming là kiểu lập trình lấy đối tượng làm nền tảng
- Đơn giản hóa việc phát triển chương trình
- Tạo ra các chương trình có tình mềm dẻo và linh động cao
- Dễ dàng phát triển, bảo trì và nâng cấp.

# Phân biệt Class & Object

- Class chỉ một cái gì đó chung chung, object là một cái cụ thể
- Ví dụ:
  - Công thức làm bánh quy là một Class → bánh quy là một Object
  - Con gái là một Class → Hồng là một Object
  - Con mèo là một Class → Con mèo Mimi nhà tôi là một Object
  - Bản vẽ là một Class → Ngôi nhà của tôi là một Object
- Hướng dẫn khai báo và sử dụng class ?

## Phần 02: Property & Method

# Phương thức và thuộc tính của đối tượng

## Thuộc tính (Property)

- Là các đặc điểm, đặc tính của một lớp

## Phương thức (Method)

- Là các hành động có thể được thực hiện từ lớp
- Phương thức cũng giống như hàm, nhưng là hàm riêng của lớp

## Ví dụ về phương thức và thuộc tính của đối tượng

### Xét Object “Bạn gái Hồng”

- Đặc điểm: tóc dài, má lúng đồng tiền, cao 1.7 m, ... → thuộc tính
- Hành động: ăn, ngủ, dạy anh văn, đánh đàn, ... → phương thức

### Xét Object “Con mèo Mimi nhà tôi”

- Đặc điểm: lông xoăn, màu xám, đuôi ngắn, ... Hành động: ăn, ngủ, bắt chuột, ....

### Xét Object “Con mèo Kitty nhà bạn”

- Đặc điểm: màu trắng, đuôi dài, ... Hành động: ăn, ngủ, làm nũng, ...

## Xây dựng Class với các phương thức và thuộc tính

- Chúng ta đã học qua các kiểu dữ liệu như INT, STRING, FLOAT để mô tả số nguyên, chuỗi và số thực → có kiểu dữ liệu mới → mô tả kiểu dữ liệu mới bằng cách nào → định nghĩa kiểu dữ liệu mới này thông qua Class
- Trước khi xây dựng một Class, chúng ta cần xác định 2 vấn đề sau:
  - Thuộc tính
  - Phương thức
- Xây dựng Class ConMeo với các phương thức và thuộc tính của nó !

## Phần 03: Constructor & Destructor

## Phương thức \_\_construct()

- \_\_construct() được gọi tự động và được gọi đầu tiên khi một object được khởi tạo.
- Thường dùng để khởi tạo các giá trị ban đầu, các trường hợp gọi \_\_construct()
  - \_\_construct()
  - \_\_construct() với tham số
  - \_\_construct() với tham số mặc định
  - \_\_construct() với cách đặt tên trùng với tên class
  - \_\_construct() với tham số là mảng

## Phương thức \_\_destruct()

- \_\_destruct() là phương thức tự động chạy khi đối tượng được khởi tạo. Nó chỉ được thực thi ở cuối trang mà đối tượng được tạo ra
- Phương thức này thường dùng để tạo hoặc hủy một session, giải phóng bộ nhớ, đóng kết nối của ứng dụng đến database, đóng kết nối đến tập tin, ...

## Phần 04: Tính kế thừa & overwrite

## Tìm hiểu từ khóa FINAL

- PHP5 cho phép định nghĩa class và method với từ khóa FINAL
- Đối với method: Lớp con không thể override các phương thức ở lớp cha nếu các phương thức ở lớp cha có khóa FINAL
- Đối với class: khi chúng ta dùng từ khóa FINAL thì chúng ta không thể extends từ class đó

## Phần 05: Phạm vi và sự ảnh hưởng

## Public – Private – Protected

- Public: có thể truy cập từ mọi nơi
- Protected: chỉ sử dụng cho class đó và các class được kế thừa từ class đó
- Private: chỉ sử dụng ở chính class đó

# Static

- Được dùng với phương thức và biến
- Truy cập nhanh phương thức mà không cần khởi tạo đối tượng

## Self & Parent

**self**

- Là đại diện cho cách khởi tạo lớp hiện thời và thường được sử dụng gọi đến biến số có khóa static hay hàm nào đó trong lớp hiện tại

**parent**

- Là đại diện của lớp cha và thường được sử dụng gọi đến biến số có khóa static hay phương thức nào đó trong lớp cha của lớp hiện tại

## const (Constant)

- Định nghĩa các biến có giá trị không thay đổi bằng từ khóa const
- Để truy cập vào lấy giá trị chúng ta sử dụng toán tử ::

## Phần 06: Một số phương thức khác

# clone()

- Sao chép một đối tượng từ đối tượng khác

## \_\_autoload()

- Tự động nạp các lớp từ một đường dẫn chứa các lớp được truyền vào

## \_\_sleep & \_\_wakeup()

- Phương thức `__sleep()` được thực hiện khi chúng ta đưa một đối tượng vào hàm `serialize`. Lúc này ở phương thức `__sleep()` chúng ta sẽ khai báo những thành phần nào của đối tượng được chuyển thành một chuỗi đặc biệt
- Phương thức `__wakeup()` dùng để làm sạch đối tượng như khi mới khởi tạo ban đầu với các giá trị mặc định được gán vào. Điều này giúp chúng ta không phải khởi tạo đối tượng một lần nữa

## **\_\_toString()**

- Phương thức \_\_toString() giúp chuyển đối tượng thành chuỗi

## \_\_set & \_\_get()

- Phương thức \_\_set() thiết lập các thuộc tính cho đối tượng cho dù thuộc tính đó chưa được khai báo trong class
- Phương thức \_\_get() lấy giá trị của một thuộc tính nào đó trong class cho dù thuộc tính đó được gán khóa private hoặc protected

## Abstract class

- Abstract class là class chứa bộ khung cơ bản. Nó chứa các thuộc tính và các phương thức nhưng các thành phần này chưa hoàn thiện.
- Các lớp extends từ lớp abstract sẽ có nhiệm vụ hoàn thiện các thành phần chưa đầy đủ này
- Abstract class thường được xây dựng bởi kỹ sư thiết kế hệ thống có kinh nghiệm, có các kiến thức mở rộng ngoài chuyên môn

## Interface class

- Interface class là một phần rất hữu ích cho việc định nghĩa một API chuẩn và đảm bảo tất cả các nhà cung cấp hoàn thiện nó

# Chương 06

# PHP Extensions

## PHP Extensions

- PHP Extensions là tên gọi được tác giả đặt ra để chỉ tập hợp các class PHP hay và hữu dụng được chia sẻ miễn phí trên Internet

# Class 01: PHPTHUMB

- PHPTHumb là một thư viện giúp chúng ta xử lý các vấn đề liên quan đến hình ảnh: thay đổi kích thước hình ảnh, xoay hình ảnh, ...
- Download: <http://phpthumb.gxdlabs.com/>

## Class 02: PHPMailer

- **Cài đặt server mail** tốn thời gian nhưng lại không sử dụng được nhiều (chủ yếu được dùng để test ở localhost)
- **Cấu hình server để gửi mail qua SMTP** cách này bị hạn chế vì trên host hạn chế việc cấu hình server. Thông thường, các host ở các nhà cung cấp của Việt Nam thì hàm mail() đều bị khóa
- **Sử dụng thư viện PHPMailer** để gửi mail thông qua SMTP → chúng ta có thể gửi mail ở bất kì đâu như localhost hay host bị khóa hàm mail()

# Class 03: FCK Editor

- FCK Editor - một ứng dụng soạn thảo văn bản trên nền web được cung cấp hoàn toàn miễn phí
- Download: <http://sourceforge.net/projects/fckeditor/>
- Cấu hình
- Cài đặt Plugin
  - File Manager
  - FLV Player

# Class 03: FCK Editor

## 1. Chuyển mã HTML

```
FCKConfig.HtmlEncodeOutput = false ;
```

## 2. Định dạng font chữ Unicode

```
FCKConfig.ProcessHTMLEntities = true ;
```

## 3. Tạo khoảng cách cho phím tab

```
FCKConfig.TabSpaces = 0 ;
```

# Class 03: FCK Editor

## 4. Thẻ HTML khi xuống dòng

ENTER: FCKConfig.HtmlEncodeOutput = false ;

SHIFT + ENTER: FCKConfig.ShiftEnterMode = 'br'

## 5. Loại bỏ thẻ HTML không cần thiết

FCKConfig.RemoveFormatTags = 'b,big,code' ;

## 6. Thay đổi skin

FCKConfig.SkinPath = FCKConfig.BasePath + 'skins/office2003/'  


# Class 03: FCK Editor

## 7. Thay đổi các thành phần trong toolbar

```
FCKConfig.ToolbarSets ["Default"]
```

## 8. Thay đổi ngôn ngữ

```
FCKConfig.AutoDetectLanguage = false;
```

```
FCKConfig.DefaultLanguage = 'vi';
```

# FileManger Plugin

## 1. Thay đổi đường dẫn

FCKConfig.LinkBrowserURL

FCKConfig.ImageBrowserURL

FCKConfig.FlashBrowserURL

## 2. Thay đổi kích thước cửa sổ

FCKConfig.LinkBrowserWindowWidth

FCKConfig.LinkBrowserWindowHeight

FCKConfig.ImageBrowserWindowWidth

FCKConfig.ImageBrowserWindowHeight

FCKConfig.FlashBrowserWindowWidth

FCKConfig.FlashBrowserWindowHeight

# FileManger Plugin

## 3. Đóng các chức năng của FCK Filemanager

```
FCKConfig.LinkUpload      = false  
FCKConfig.ImageUpload     = false  
FCKConfig.FlashUpload     = false
```

## 4. Thay đổi thư mục upload (/fckeditor/editor/plugins/ajaxfilemanager/inc/config.base.php)

```
define('CONFIG_SYS_DEFAULT_PATH', ' ../../../../../../uploaded/' );  
define('CONFIG_SYS_ROOT_PATH', ' ../../../../../../uploaded/' );
```

## 5. Bật chế độ bảo mật cho phần upload

```
define('CONFIG_ACCESS_CONTROL_MODE', false);  
define("CONFIG_LOGIN_USERNAME", 'sfb5e');  
define('CONFIG_LOGIN_PASSWORD', 'sdjfjhkjxcf');
```

# FLV Player Plugin

## 1. Khai báo plugin

```
FCKConfig.Plugins.Add( 'flvPlayer' ) ;
```

## 2. Gắn FLV vào toolbar

## 3. Chỉnh sửa hàm BrowseServer (fckeditor/editor/plugins/flvPlayer/flvPlayer.js)

MediaBrowserURL

= FlashBrowserURL

MediaBrowserWindowWidth

= FlashBrowserWindowWidth

MediaBrowserWindowHeight

= FlashBrowserWindowHeight

## 4. Chỉnh hiển thị nút nhấn

```
GetE('tdBrowse').style.display = oEditor.FCKConfig.FlashBrowser ? '' : 'none' ;
```

## Class 04: Captcha

- Completely Automated Public Turing test to tell Computers and Humans Apart
- Captcha là 1 hình ảnh dùng để phân biệt giữa người và chương trình máy tính. Captcha dùng để xác định liệu có phải là một người dùng đang truy cập hệ thống hay một chương trình máy tính đang truy cập hệ thống

# Class 04: Class Securimage

- **Cài đặt và sử dụng Securimage** (<http://www.phpcaptcha.org/download/>)
- **Tìm hiểu một số cách cấu hình**
  1. Thay đổi màu sắc
  2. Sử dụng background
  3. Độ phức tạp của captcha
  4. Thay đổi font chữ

# Class 04: Class Securimage

- **Tìm hiểu một số cách cấu hình**

5. Chữ ký

6. Kích thước hình ảnh

7. Ký tự xuất hiện trong captcha

- **Refresh captcha**

# Class 05: SimplePie

- **RSS**
  - RSS là định dạng dữ liệu dựa theo chuẩn XML được sử dụng để chia sẻ và phát tán nội dung của một website nào đó.
- **SimplePie (<http://simplepie.org/>)**
  - SimplePie là một class PHP hoàn toàn miễn phí giúp việc thao tác, làm chủ các RSS dễ dàng và nhanh chóng hơn.

# Tìm hiểu về OpenID

## KHÁI NIỆM

- OpenID là một giải pháp đăng nhập, giúp cho người sử dụng có thể đăng nhập các trang web khác nhau (hỗ trợ OpenID) chỉ bằng một tài khoản duy nhất
- Các trang web lớn hiện đang cung cấp dịch vụ OpenID đáng để nhắc đến đó là Google, Facebook, Yahoo!, Microsoft, AOL, MySpace, ...

# Tìm hiểu về OpenID

## MẶT TÍCH CỰC VÀ HẠN CHẾ

- Tạo hướng tiếp cận truy cập mới cho người dùng, các quá trình thực thi trên website diễn ra nhanh hơn vì đã thông qua bước xác nhận tài khoản bằng OpenID
- Tiết kiệm thời gian xây dựng phần quản lý người sử dụng
- Không có toàn quyền kiểm soát tài khoản của người sử dụng

## Class 06: LightOpenId

- Thư viện LightOpenID cung cấp tất cả những vấn đề cần thiết để chúng ta có thể làm việc và tương tác với OpenID ngay trên website PHP của chúng ta
- Download <http://gitorious.org/lightopenid>

## Class 07: Mobile Detect

- Class gọn nhẹ và miễn phí giúp chúng ta nhận dạng thiết bị đang truy cập vào website của chúng ta
- Download <http://mobiledetect.net/>

# Vấn đề

## QUÁ TRÌNH TIẾP NHẬN VÀ HIỂN THỊ NỘI DUNG TRANG WEB

- Người sử dụng gửi yêu cầu truy cập trang web
- Server tiếp nhận, phân tích và xử lý yêu cầu
- Server chuyển dữ liệu về cho người sử dụng

→ việc phân tích xử lý yêu cầu (truy vấn database) sẽ làm chậm tốc độ thực thi của website → giải pháp ?

# Cache

- Cache được sử dụng như một trong nhiều giải pháp giúp tăng tốc độ của website (hạn chế việc truy cập vào database)
- Các kiểu cache thường gặp
  - Cache toàn bộ page
  - Cache từng phần của page
  - Cache SQL

# Các hàm thao tác với cache

- ob\_start()
- ob\_get\_contents()
- ob\_clean()
- ob\_end\_flush()

# Class 09: Validate

# Chương 07: Ngôn ngữ SQL

# MySQL

- MySQL là một hệ quản trị cơ sở dữ liệu mã nguồn mở.
- MySQL thường được kết hợp cùng ngôn ngữ lập trình PHP trong việc triển khai và xây dựng các website
- Cài đặt MySQL ?

# Thiết kế CSDL Quản lý công trình

## THỰC TRẠNG

- Một công trình với một kinh phí nhất định được phụ trách bởi một chủ đầu tư và được xây dựng cho một chủ sở hữu nào đó.
- Công trình được chủ đầu tư thuê một hoặc nhiều kiến trúc sư để thiết kế và thuê một hoặc nhiều công nhân để tiến hành xây dựng.
- Một kiến trúc sư có thể thiết kế một hoặc nhiều công trình cùng một lúc. Khi kiến trúc sư thiết kế một công trình sẽ có một thù lao tương ứng
- Một công nhân có thể tham gia xây dựng một hoặc nhiều công trình cùng một lúc. Lúc này sẽ được ghi nhận lại ngày tham gia và số ngày đã tham gia

# Thiết kế CSDL Quản lý công trình

## XÂY DỰNG CƠ SỞ DỮ LIỆU

- **Architect** ( id, name, birthday, sex, place, address )
- **Contractor** ( id, name, address )
- **Host** ( id, name, address )
- **Worker** ( id, name, birthday, year, skill )
- **Building** ( id, name, address, city, cost, start, *contractor\_id*, *host\_id* )
- **Work** ( building\_id, worker\_id, date, total )
- **Design** ( building\_id, architect\_id, benefit )

# Ngôn ngữ SQL

## ĐỊNH NGHĨA

- SQL (Structured Query Language) là một loại ngôn ngữ máy tính phổ biến để tạo, sửa, và lấy dữ liệu từ một hệ quản trị cơ sở dữ liệu quan hệ.

## Ngôn ngữ SQL

### SQL GỒM 2 THÀNH PHẦN CHÍNH

- DDL (Data Definition Language) định nghĩa cấu trúc của CSDL.
  - Cấu trúc của CSDL: CREATE TABLE, CREATE VIEW, ALTER TABLE,...
  - Điều khiển quyền truy cập trên dữ liệu: GRANT, REVOKE,...
- DML (Data Manipulation Language) truy xuất và cập nhật dữ liệu: INSERT, UPDATE, DELETE, SELECT

## Ngôn ngữ SQL – Cú pháp

- Chữ hoa đại diện cho từ khóa.
- Chữ thường đại diện cho các từ của người dùng định nghĩa
- Dấu | chỉ sự lựa chọn.
- Dấu { chỉ phần tử bắt buộc phải có.
- Dấu [ chỉ phần tử tùy chọn (không bắt buộc).
- Dấu ... chỉ thành phần có thể lặp lại từ 0 đến nhiều lần.

## Ngôn ngữ SQL – Lưu ý

- Hầu hết các phần trong câu lệnh SQL là không phân biệt chữ hoa chữ thường, trừ các ký tự trong chuỗi dữ liệu.
- Nên viết các từ khóa của SQL bởi ký tự chữ hoa và các từ do người dùng định nghĩa (table, column, ...) bởi ký tự chữ thường

# Ngôn ngữ SQL

## CƠ SỞ DỮ LIỆU QUẢN LÝ PHÒNG

PHONG ( ma\_phong, ten\_phong, truong\_phong )

*Phòng có một mã số duy nhất, một tên phòng, trưởng phòng là mã nhân viên của trưởng phòng.*

NHANVIEN ( ma\_nv, hten\_nv, phai, cviec, luong, phu\_cap, ma\_phong )

*Nhân viên có một mã duy nhất, họ tên, phái, công việc, lương, phụ cấp và thuộc một phòng nào đó.*

TĐO\_NN ( ma\_nv, ngoaingu, bang\_cap )

*Một nhân viên có thể biết nhiều ngoại ngữ, mỗi ngoại ngữ có thể có các bảng cấp khác nhau.*

## Truy vấn đơn giản

```
SELECT [ DISTINCT | ALL ]
      { * | <tên cột 1> [, <tên cột 2> [, ...] ] }
  FROM <tên bảng>
```

- DISTINCT      Hiển thị những dòng phân biệt.
- ALL:           hiển thị tất cả các dòng (giá trị mặc định)
- \*                thay thế tất cả các cột trong bảng.

## SẮP XẾP KẾT QUẢ

```
ORDER BY <Tên Cột | STT Cột> [ASC| DESC]  
[,<Tên Cột | STT Cột> [ASC| DESC],...]
```

- ASC Sắp xếp theo kết quả tăng dần (giá trị mặc định)
- DESC Sắp xếp theo kết quả giảm dần

→ example 04

## Aggregate Functions

- AVG Xác định giá trị trung bình
- SUM Xác định giá trị tổng
- MIN Xác định giá trị nhỏ nhất
- MAX Xác định giá trị lớn nhất
- COUNT Xác định tổng số phần tử

## Truy vấn con – Định nghĩa

- Nếu điều kiện chọn ở mệnh đề WHERE cần truy cập thông tin ở các bảng khác với bảng đang truy vấn để kiểm tra điều kiện ta sẽ sử dụng một câu select khác lồng trong điều kiện ở mệnh đề WHERE

## Truy vấn con – Các dạng thường gặp

- Dạng 01: <Tên cột> <so sánh> (<select con>) *điều kiện đúng khi giá trị của cột so sánh đúng với giá trị trả về từ select con*
- Dạng 02: Tên cột> <so sánh> ALL (<select con>) *điều kiện đúng khi giá trị của cột so sánh đúng với tất cả các giá trị trả về từ select con*
- Dạng 3: <Tên cột> <so sánh> ANY|SOME (<select con>) *điều kiện đúng khi giá trị của cột so sánh đúng với bất kỳ một giá trị nào trả về từ select con*

## Truy vấn con – Các dạng thường gặp

- Dạng 04: <Tên cột> [NOT] IN (<select con>) *điều kiện đúng khi giá trị của cột nằm trong tập hợp các giá trị trả về của select con*
- Dạng 05: [NOT] EXISTS (<select con>) *điều kiện đúng khi kết quả trả về của select con khác rỗng.*

## Truy vấn con – Các quy luật

- Mệnh đề ORDER BY không được sử dụng trong truy vấn con (mặc dù chúng có thể được dùng trong truy vấn cha).
- Mệnh đề SELECT trong truy vấn con chỉ được bao gồm một cột duy hoặc 1 biểu thức duy nhất, ngoại trừ truy vấn con sử dụng EXISTS.
- Truy vấn con có thể truy cập các cột của các bảng ở truy vấn cha bằng cách sử dụng bí danh

## Truy vấn con – Kết quả trả về

- Dạng 01: Một cột và một dòng
- Dạng 02: Một cột và nhiều dòng
- Dạng 03: Một cột và nhiều dòng
- Dạng 04: Một cột và nhiều dòng
- Dạng 05: Nhiều cột và nhiều dòng

# GROUP BY – HAVING

## GROUP BY

- Đặc tính gom nhóm cho phép chúng ta thực hiện các chức năng trên một nhóm các dòng như là một dòng riêng biệt.
- Sử dụng mệnh đề GROUP BY sau mệnh đề FROM hoặc WHERE

```
SELECT <các cột để phân nhóm>, <hàm-kết-tập(<bíểu thức>)>  
FROM <tên bảng> [ WHERE <Điều kiện> ]  
GROUP BY <Cột để phân nhóm> [, <Cột để phân nhóm> [, ...]]
```

## GROUP BY – HAVING

### HAVING

- Sử dụng mệnh đề HAVING theo sau mệnh đề GROUP BY để lọc ra các nhóm theo điều kiện, sau khi đã phân nhóm:

**HAVING <điều kiện chọn trên nhóm>**

## THỨ TỰ THỰC HIỆN CÁC MỆNH ĐỀ

**SELECT ... FROM ...**

**WHERE ...**

**GROUP BY ... HAVING ...**

**ORDER BY ...**

FROM → WHERE → GROUP BY → HAVING → SELECT → ORDER BY

## NHẬN BIẾT THÔNG BÁO LỖI

1. Tên cột không tìm thấy: *Invalid column name ...*
2. Tên bảng không tìm thấy: *Invalid object name ...*
3. Lỗi cú pháp: *Incorrect Syntac near Incorrect Syntac near*
4. Tên cột có mặt ở nhiều bảng: *Ambiguous column name*
5. Phép so sánh không tương thích kiểu: *Error converting data type*

## KẾT NỐI NGOẠI – OUTER JOIN

- Nếu một dòng của 1 bảng được kết nối không khớp với dòng nào bên bảng kia, thì dòng đó sẽ không xuất hiện ở bảng kết quả
- Kết nối ngoại cho phép giữ lại những dòng không thỏa điều kiện kết nối

## KẾT NỐI NGOẠI – OUTER JOIN

- Có 3 loại kết nối
  1. FROM <Table1> LEFT JOIN <Table2> ON <condition>
  2. FROM <Table1> RIGHT JOIN <Table2> ON <condition>
  3. FROM <Table1> JOIN <Table2> ON <condition>

# Xử lý chuỗi trong MySQL

| STT | Tên hàm                        | Chức năng  |
|-----|--------------------------------|--|
| 1   | LENGTH (str)                   | Xác định chiều dài chuỗi str   |
| 2   | CONCAT (str1, str2, ..., strn) | Nối các chuỗi str1, str2, ..., strn thành một chuỗi duy nhất   |
| 3   | FORMAT (str, n)                | Chuyển đổi định dạng chuỗi str theo dạng #,###,###.##, trong đó làm tròn đến n số thập phân          |
| 4   | INSERT (str, pos, len, newstr) | Thay thế các ký tự trong chuỗi str từ vị trí thứ pos đến vị trí thứ pos+len-1 bởi chuỗi ký tự newstr |
| 5   | INSTR (str, substr)            | Trả về vị trí xuất hiện đầu tiên của chuỗi substr trong chuỗi str                                    |
| 6   | LEFT (str, len)                | Trả về chuỗi ký tự tính từ vị trí bên trái của chuỗi str từ vị trí thứ nhất đến vị trí thứ len       |

# Xử lý chuỗi trong MySQL

| STT | Tên hàm                   | Chức năng   |
|-----|---------------------------|---|
| 7   | RIGHT (str, len)          | Trả về chuỗi ký tự tính từ vị trí bên phải của chuỗi str từ vị trí thứ nhất đến vị trí thứ len. |
| 8   | LOCATE (substr, str, pos) | Trả về vị trí xuất hiện đầu tiên của chuỗi substr trong chuỗi str tính từ vị trí thứ pos        |
| 9   | LOWER (str)               | Chuyển đổi chuỗi str thành chữ thường   |
| 10  | UPPER (str)               | Chuyển đổi chuỗi str thành chữ hoa  |
| 11  | LTRIM (str)               | Loại bỏ tất cả khoảng trắng bên trái chuỗi str  |
| 12  | RTRIM (str)               | Loại bỏ tất cả khoảng trắng bên phải chuỗi str  |

# Xử lý chuỗi trong MySQL

| STT | Tên hàm  | Chức năng   |
|-----|--|---|
| 13  | REPLACE (str, from_str, to_str)  | Thay thế các ký tự from_str trong chuỗi str bởi các ký tự to_str  |
| 14  | REVERSE (str)  | Đảo ngược chuỗi str   |
| 15  | SUBSTRING (str FROM pos FOR len)   | Trích lọc từ chuỗi str một chuỗi mới, chuỗi mới này có các ký tự bắt đầu từ vị trí pos (tính từ vị trí biên trái) cho đến vị trí pos+len-1                  |
| 16  | TRIM ({{BOTH   LEADING   TRAILING} [remstr] FROM} str),<br>TRIM([remstr FROM] str) | Trả về chuỗi str với các ký tự remstr đã bị loại bỏ, theo các tiêu chí LEADING (bên trái), TRAILING (bên phải) và BOTH (cả hai vị trí bên trái và bên phải) |

# Regular expression

- Sử dụng biểu thức chính quy trong MySQL được chia thành 3 nhóm từ khóa sau:
  - NOT\_REGEXP
  - REGEXP
  - RLIKE
- Trong nội dung này trình bày việc sử dụng biểu thức chính quy với từ khóa REGEXP. Kết quả tìm kiếm sẽ trả về 1 nếu tìm thấy hoặc trả về 0 nếu không tìm thấy.

| TT | Ký hiệu | Diễn giải  |
|----|---------|--|
| 1  | ^       | Tìm từ đầu chuỗi nguồn   |
| 2  | \$      | Tìm từ cuối chuỗi nguồn  |
| 3  | .       | Đại diện một ký tự bất kỳ  |
| 4  | *       | Ký tự xuất hiện 0 hoặc nhiều lần   |
| 5  | +       | Ký tự xuất hiện 1 hoặc nhiều lần   |
| 6  | ?       | Ký tự xuất hiện 0 hoặc 1 lần   |
| 7  |         | Sự lựa chọn (hoặc)   |
| 8  | {n.m}   | Số lần xuất hiện của ký tự (a* tương đương a{0}; a+ tương đương a{1}; a? tương đương a{0,1}) |

## CẬP NHẬT CƠ SỞ DỮ LIỆU

- Các lệnh cập nhật CSDL của SQL
  - INSERT
  - DELETE
  - UPDATE

# Chương 08: PHP & MySQL

## Kết nối – Hủy kết nối

- `mysql_connect()` kết nối đến hệ quản trị cơ sở dữ liệu MySQL
- `mysql_error()` ghi nhận lỗi trong quá trình kết nối
- `mysql_select_db()` lựa chọn cơ sở dữ liệu muốn thao tác
- `mysql_close()` đóng kết nối

## Insert – Update – Delete

- `mysql_query()` thực thi câu SQL
- `mysql_affected_rows()` trả về số dòng đã được insert, update, delete

# Truy xuất dữ liệu

# SQL Injection

## KHÁI NIỆM

- SQL injection - kỹ thuật cho phép thực hiện những câu SQL bất hợp pháp dựa vào lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các ứng dụng web và các thông báo lỗi của hệ quản trị cơ sở dữ liệu
- Thường xảy ra trên các ứng dụng web có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, Oracle, DB2, Sybase.

# SQL Injection

## CÁCH TẤN CÔNG CƠ BẢN

```
SELECT * FROM user WHERE name = [userName] ;
```

→ điều gì xảy ra nếu **[userName]** mang giá trị:

- johnsmith OR “a” = “a”
- johnsmith; DROP TABLE user; SELECT \* FROM data WHERE “a” = “a”

# XÂY DỰNG CLASS THAO TÁC VỚI DATABASE

## Exercise 01 – Login website

## Exercise 02 – User online

## Exercise 02 – User online

### MIÊU TẢ

|                   |                     |                          |
|-------------------|---------------------|--------------------------|
| • Page: index.php | Info: userA – 19h00 | Online: 1 (userA)        |
| • Page: index.php | Info: userB – 19h10 | Online: 2 (userA, userB) |
| • Page: index.php | Info: userB – 19h20 | Online: 1 (userB)        |

(quy định trong 15 phút nếu người dùng không có thao tác gì đối với trang hiện tại xem như không online ở trang đó)

## Exercise 02 – User online

### HƯỚNG GIẢI QUYẾT

- Cơ sở dữ liệu: online (id, ip, url, time)
- Khi người dùng truy cập một trang nào đó, xử lý
  - Tìm kiếm thông tin người dùng trong bảng online
    - Nếu có: Cập nhật lại cột time
    - Nếu không có: thêm mới
  - Xóa các dòng dữ liệu có time không phù hợp với thời gian quy định
  - Hiển thị danh sách các người dùng online tại trang đó

## Exercise 03 – Manage User

# Pagination

# Chương 09: PHP & AJAX

Lập trình PHP  
zend.vn

# Phần 01: JSON

# JSON

- JSON (JavaScript Object Notation) là một đối tượng trung gian giúp việc trao đổi thông tin giữa các trình duyệt, giữa các ngôn ngữ lập trình khác nhau, ... trở nên dễ dàng hơn.

## Phần 02: jQuery & Ajax

## Ajax là gì ?

- AJAX, viết tắt từ Asynchronous JavaScript and XML, kỹ thuật này cho phép tăng tốc độ ứng dụng web bằng cách cắt nhỏ dữ liệu và chỉ hiển thị những gì cần thiết, thay vì tải đi tải lại toàn bộ trang web.
- Kiến thức cần có: HTML, CSS, JavaScript, Jquery, Json, PHP

```
.load (url [,data] [,complete(responseText, textStatus)])
```

```
jQuery.get (url [, data] [, success(data, textStatus)[,  
dataType]])
```

```
jQuery.post (url [, data] [, success(data, textStatus)[,  
          dataType]])
```

# jQuery.ajax(setting)

## Phần 04: jQuery Form Plugin

# Chương 10:

# Xây dựng ứng dụng

# Tìm hiểu mô hình MVC

## Mô hình MVC – Khái niệm

- Trong một ứng dụng PHP chúng ta thường đặt các mã PHP, câu truy vấn, phần giao diện, ... trên cùng một trang
- Điều này làm cho việc sửa chữa và nâng cấp trở nên tốn thời gian vì vậy cấu trúc ứng dụng bây giờ thường được chuyển sang mô hình MVC (Model – View - Controller)

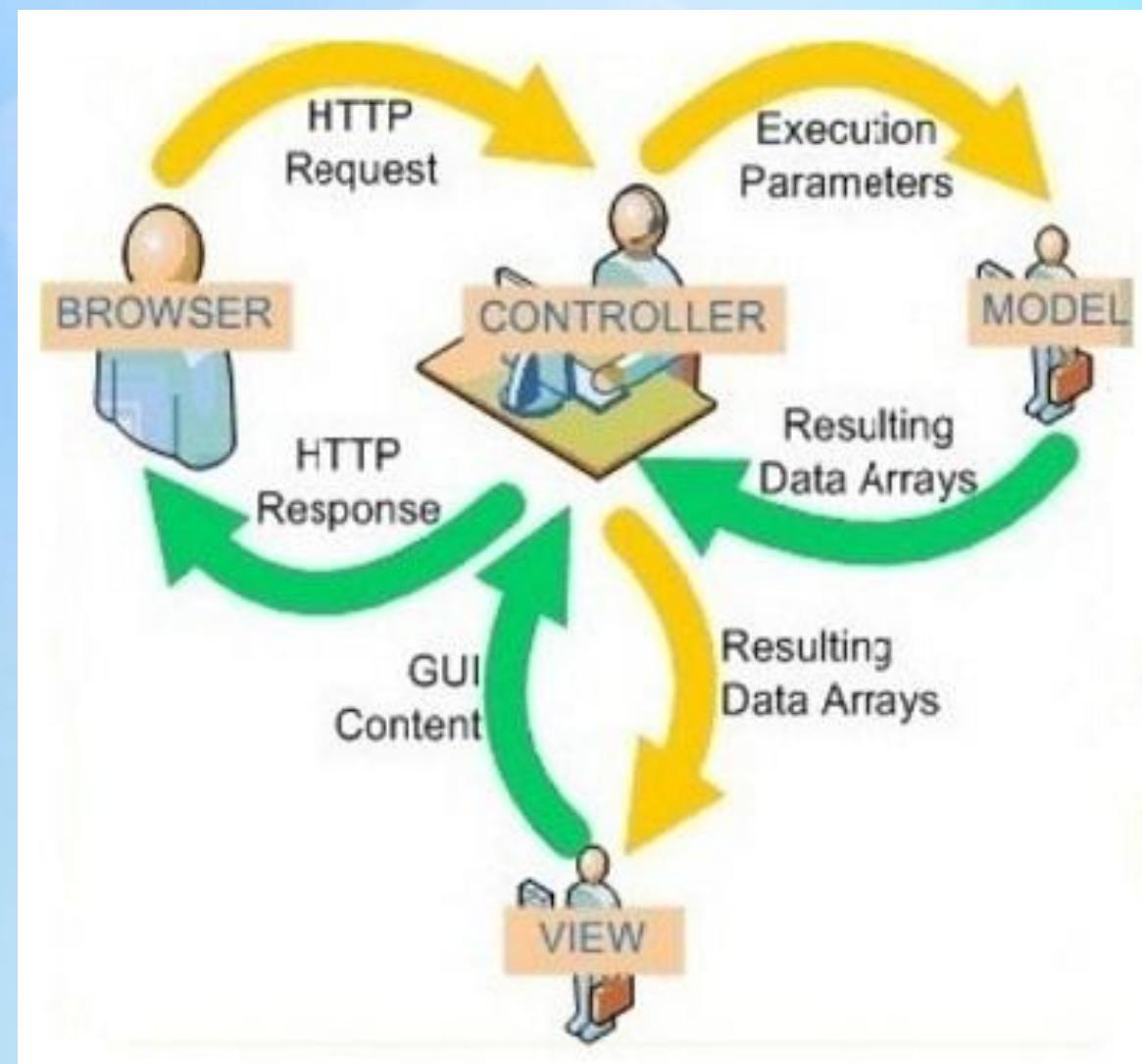
## Mô hình MVC – Khái niệm

- Mô hình MVC sẽ chia một chức năng nào đó thành 3 phần:
  - Tương tác dữ liệu (Model)
  - Xử lý mã nguồn (Controller)
  - Hiển thị dữ liệu (View)

## Mô hình MVC – Chức năng các thành phần

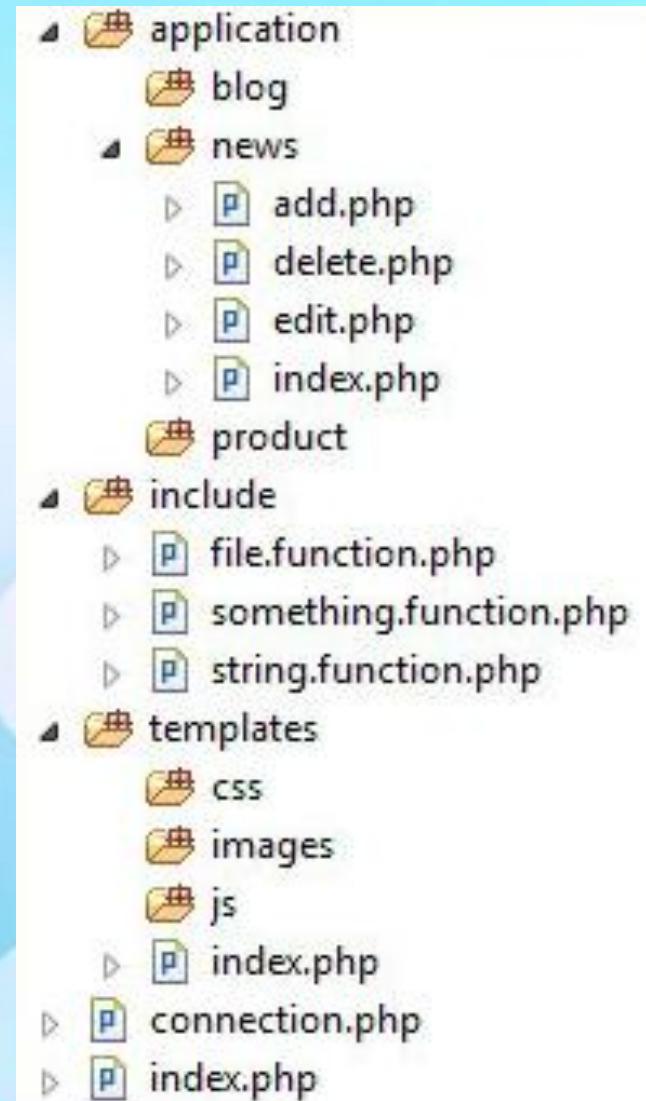
- Model: thao tác với cơ sở dữ liệu (connect, insert, update, select, ...)
- View: tạo ra giao diện hiển thị dữ liệu
- Controller: điều hướng các nhiệm vụ đến đúng phương thức có chức năng xử lý nhiệm vụ đó

# Mô hình MVC – Luồng xử lý



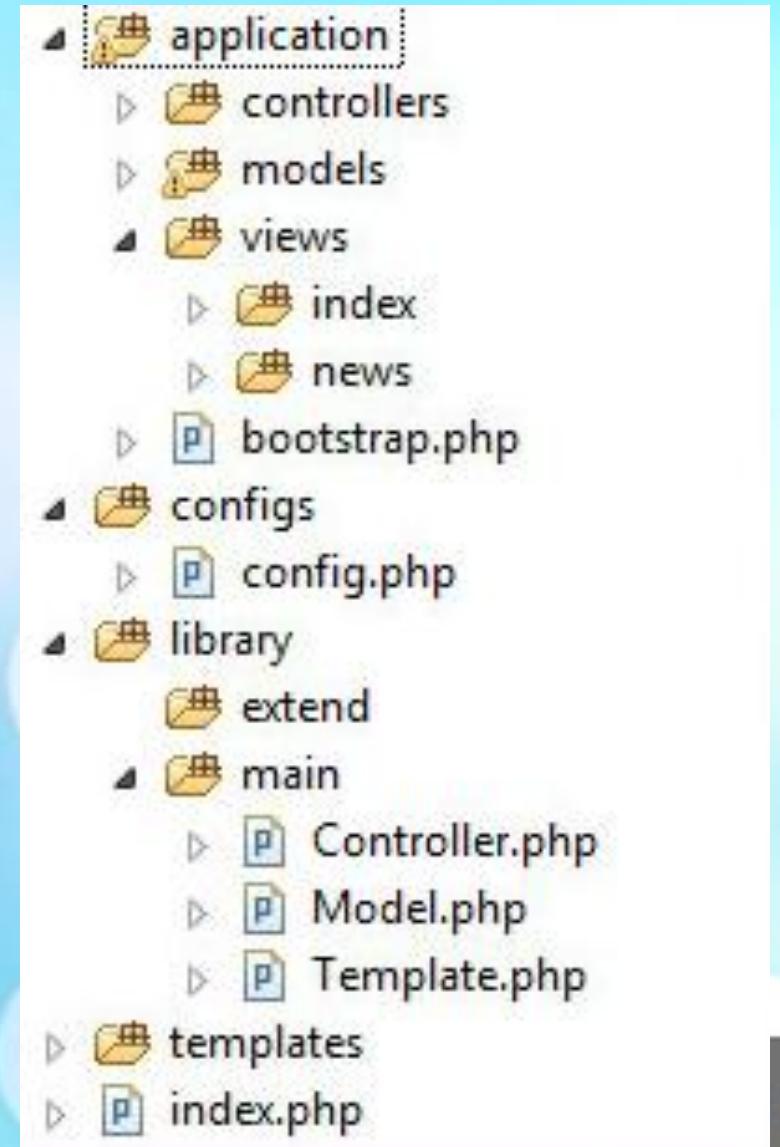
# Cấu trúc thư mục mô hình thông thường

- Một tập tin chạy chính (index.php)
- Một tập tin kết nối đến cơ sở dữ liệu (connection.php) và được nhúng vào index.php
- Một thư mục chứa giao diện (template)
- Một thư mục chứa các hàm xử lý (include)
- Một thư mục chứa các tập tin xử lý cho từng chức năng (application)



# Cấu trúc thư mục mô hình MVC

- Thư mục chứa các thành phần cơ bản của MVC (application)
- Thư mục chứa các tập tin cấu hình của ứng dụng (configs)
- Thư mục chứa thư viện của các class (library)
  - Controller.php điều hướng chức năng
  - Model.php xử lý dữ liệu
  - Template.php xử lý giao diện
- Thư mục chứa giao diện của ứng dụng (templates)
- Tập tin nạp các lớp điều khiển của ứng dụng (bootstrap.php)
- Tập tin chạy chính (index.php)



## Mô hình MVC – Ưu nhược điểm

- **Ưu điểm:**
  - Xây dựng và phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp.
  - Khâu bảo trì và nâng cấp đơn giản, không ảnh hưởng quá nhiều đến các thành phần khác
- **Nhược điểm:**
  - Đối với dự án nhỏ việc áp dụng mô hình MC gây cồng kềnh, tốn thời gian trong quá trình phát triển.
  - Tốn thời gian trung chuyển dữ liệu của các tầng

# MVC - Multy module & Multy template

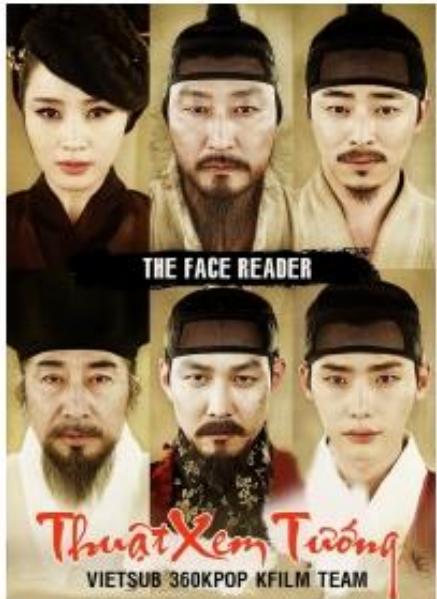
# Lập trình PHP

*(bổ sung một số chuyên đề trong khóa lập trình PHP của ZendVN)*

# Chuyên đề bổ sung

1. Tạo file PDF với thư viện mPDF
2. Kỹ thuật trình bày dữ liệu với Ajax
3. Kỹ thuật phân trang bằng Ajax
4. Lấy tin tự động bằng DOM
5. Sử dụng .htaccess trong ứng dụng web

# Kỹ thuật hiển thị dữ liệu bằng ajax



The Face Reader

2013



The Pirate Fairy

2014



Sadako

2013



Resident Evil 3

2007

Xem thêm

Lập trình PHP  
zend.vn

# Kỹ thuật phân trang bằng ajax

## Khoá học trực tuyến



### Khóa học lập trình jQuery (jQuery Master)

Nếu nói đến lập trình web, chúng ta nghĩ ngay đến một ngôn ngữ lập trình không thể thiếu trong một website đó là Javascript. Javascript giúp chúng ta thực hiện những thao tác, hiệu ứng mà các ngôn ngữ lập web như PHP, ASP, JSP... khó mà thực hiện được.



### Xây dựng web chat với NodeJS

Khóa học jQuery này cũng bổ xung thêm cho các bạn một Javascript framework khác đó là NodeJS qua một ví dụ thực tế là ứng dụng Web chat. Với kiến thức và ví dụ của NodeJS các bạn có thể dễ dàng làm phần Chat support trên website của chính mình



### Khóa học lập trình PHP

PHP đã và đang trở thành một ngôn ngữ lập trình phổ biến trong các ứng dụng web ngày nay. Với các ưu điểm như miễn phí, dễ đọc, thao tác tốt với các hệ quản trị cơ sở dữ liệu, cộng đồng sử dụng rộng rãi, thư viện phong phú ... PHP ngày càng phát triển và việc học PHP đã trở nên đặc biệt cần thiết đối với bất kỳ lập trình

# Lấy tin tự động bằng DOM

VnExpress

DanTri



## Trọng tài lo tuyển nữ bị ép trong trận gặp Thái Lan

Trước trận play-off tranh vé dự World Cup 2015, trọng tài từng ba lần giành danh hiệu "Còi vàng" Dương Văn Hiền có lý do để băn khoăn về chuyện người cầm cân nảy mực.



## Costa cầu cứu 'thần y' Marijana

Tiền đạo của Atletico Madrid đã đến Serbia hôm qua để gặp nữ bác sĩ Marijana Kovacevic - người sở hữu phương thuốc đặc biệt chiết xuất từ dạ con phụ nữ sau sinh để điều trị chấn thương.



## Những màn nội chiến ở chung kết Cup châu Âu

Trận derby Atletico - Real Madrid thứ bảy này là lần thứ năm Champions League chứng kiến hai đội cùng quốc gia tranh ngôi vô địch.



## Pirlo say khướt trong lễ rước Cup của Juventus

CLB thành Torino vừa thâu tóm chức vô địch Serie A thứ ba liên tiếp và thứ 30 trong lịch sử vẻ vang.

# Ứng dụng của tập tin .htaccess

1. Hạn chế IP truy cập vào website

6. Tạo mật khẩu cho thư mục tập tin

2. Điều hướng thông báo lỗi

7. Chống chia sẻ tài nguyên

3. Thiết lập tập tin thực thi mặc định

8. Thiết lập kích thước tập tin upload

4. Hạn chế view thư mục

9. Rewrite URL

5. Hạn chế truy cập vào tập tin

...

# KỸ THUẬT XỬ LÝ MẢNG CHUYÊN SÂU

## 1. Khái niệm:

Mảng là một biến đặc biệt và có thể lưu trữ nhiều giá trị (còn biến thì không). Trong PHP có 3 loại mảng: mảng số nguyên, mảng kết hợp và mảng đa chiều.

|              | Mảng số nguyên        | Mảng kết hợp                     | Mảng đa chiều   |
|--------------|-----------------------|----------------------------------|---|
| Tên gọi khác | Mảng liên tục         | Mảng không liên tục              | Mảng lồng   |
| Đặc điểm     | Chí số của mảng là số | Chí số của mảng là chuỗi hoặc số | Mỗi phần tử trong mảng chính có thể là một mảng và mỗi phần tử trong mảng con cũng có thể là một mảng |
| In mảng      | Câu lệnh for          | Câu lệnh foreach                 |   |

## 2. Các hàm xử lý mảng:

| STT | Hàm  | Chức năng   |
|-----|--|---|
| 1   | print_r (\$array)  | Xem cấu trúc của mảng   |
| 2   | count (\$array)  | Trả về giá trị kiểu số nguyên là số phần tử của mảng  |
| 3   | array_values (\$array)                                   | Trả về một mảng liên tục có các phần tử có giá trị là giá trị lấy từ các phần tử của mảng \$array   |
| 4   | array_keys (\$array)                                     | Trả về một mảng liên tục có các phần tử có giá trị là khóa lấy từ các phần tử của mảng \$array.   |
| 5   | array_pop (\$array)                                      | Loại bỏ phần tử cuối cùng của mảng. Hàm trả về phần tử cuối cùng đã được loại bỏ.   |
| 6   | array_shift (\$array)                                    | Loại bỏ phần tử đầu tiên của mảng. Hàm trả về phần tử đầu tiên đã được loại bỏ.   |
| 7   | array_unique (\$array)                                   | Loại bỏ những phần tử trùng nhau trong mảng và trả về mảng mới  |
| 8   | unset()  | Xóa phần tử ở vị trí bất kỳ của mảng  |
| 9   | array_push<br>(\$array, \$val1, \$val2, ... , \$valn)    | Thêm một hoặc nhiều phần tử vào cuối mảng \$array. Hàm trả về kiểu số nguyên là số lượng phần tử của mảng \$array mới   |
| 10  | array_unshift<br>(\$array, \$val1, \$val2, ... , \$valn) | Thêm một hoặc nhiều phần tử vào đầu mảng \$array. Hàm trả về kiểu số nguyên là số lượng phần tử của mảng \$array mới  |
| 11  | array_reverse (\$array)                                  | Đảo ngược vị trí các phần tử của mảng, phần tử cuối trở thành phần tử đầu tiên, phần tử kế cuối trở thành phần tử thứ nhì, ... kết quả trả về là một mảng mới |
| 12  | array_flip (\$array)                                     | Trả về một mảng có khóa và giá trị được hoán đổi cho nhau so với mảng \$array (giá trị thành khóa và khóa thành giá trị)                                      |
| 13  | array_sum (\$array)                                      | Tính tổng các phần tử trong mảng  |
| 14  | min (\$array)  | Xác định phần tử nhỏ nhất trong mảng  |

|    |  |  |
|----|--|--|
| 15 | <code>max (\$array)</code>                                   | Xác định phần tử lớn nhất trong mảng   |
| 16 | <code>array_count_values(array)</code>                       | Thống kê số lần xuất hiện của các phần tử trong mảng   |
| 17 | <code>array_merge (\$array1, \$array2, ..., \$arrayn)</code> | Nhập 2 hay nhiều mảng thành một mảng duy nhất và trả về mảng mới   |
| 18 | <code>array_rand (\$array, \$number)</code>                  | Lấy ngẫu nhiên \$number phần tử từ mảng \$array và đưa vào mảng mới (lấy giá trị khóa)   |
| 19 | <code>array_search (\$value, \$array)</code>                 | Tìm phần tử mang giá trị \$value trong mảng \$array. Trả về khóa của phần tử tìm được.   |
| 20 | <code>array_key_exists (\$key, \$array)</code>               | Kiểm tra khóa \$key có tồn tại trong mảng \$array hay không? Nếu có trả về giá trị true.   |
| 21 | <code>in_array (\$value, \$array)</code>                     | Kiểm tra giá trị \$value có tồn tại trong mảng \$array hay không? Nếu có trả về giá trị true.  |
| 22 | <code>array_slice (\$array, \$begin, \$finish)</code>        | Trích lấy 1 đoạn phần tử của mảng \$array từ vị trí \$begin đến vị trí \$finish. Phần tử đầu tiên (chỉ số 0), phần tử cuối cùng (chỉ số -1 hay count(\$array) - 1) |
| 23 | <code>array_change_key_case (\$array, case)</code>           | Chuyển đổi các key trong mảng thành chữ hoa hoặc chữ thường  |
| 24 | <code>implode (\$str, \$array)</code>                        | Chuyển các giá trị của mảng \$array thành một chuỗi bao gồm các phần tử cách nhau bởi ký tự \$str  |
| 25 | <code>explode (\$delimiter, \$str)</code>                    | Chuyển một chuỗi thành một mảng. Tách chuỗi dựa vào \$delimiter, mỗi đoàn tách ra sẽ thành một phần tử của mảng mới  |
| 26 | <code>current(\$array)</code>                                | Truy xuất phần tử hiện tại của mảng  |
| 27 | <code>end(\$array)</code>                                    | Truy xuất phần tử cuối cùng của mảng   |
| 28 | <code>next(\$array)</code>                                   | Truy xuất phần tử sau phần tử hiện tại của mảng  |
| 29 | <code>prev(\$array)</code>                                   | Truy xuất phần tử trước phần tử hiện tại của mảng  |
| 30 | <code>reset()</code>   | Quay về vị trí phần tử đầu tiên trong mảng   |
| 31 | <code>serialize (\$value)</code>                             | Chuyển chuỗi/mảng/đối tượng \$value thành một chuỗi đặc biệt để lưu vào cơ sở dữ liệu  |
| 32 | <code>unserialize (\$value)</code>                           | Chuyển chuỗi đặc biệt được tạo từ serialize(\$value) về trạng thái ban đầu   |
| 33 | <code>shuffle (\$array)</code>                               | Tạo ra mảng mới (mảng liên tục) với thứ tự các phần tử trong mảng bị thay đổi  |
| 34 | <code>compact()</code>                                       | Tạo ra mảng mới từ các biến có sẵn   |
| 35 | <code>range()</code>   | Sử dụng hàm range để tạo ra các phần tử của mảng   |
| 36 | <code>array_combine (\$keys, \$values)</code>                | Tạo một mảng mới có khóa được lấy từ mảng \$keys và giá trị được lấy từ mảng \$values theo tuần tự (Yêu cầu số phần tử ở 2 mảng phải bằng nhau)                    |
| 37 | <code>array_diff (\$array1, \$array2)</code>                 | Trả về một mảng bao gồm các phần tử có giá trị tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2   |
| 38 | <code>array_diff_key (\$array1, \$array2)</code>             | Trả về một mảng bao gồm các phần tử có khóa tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2  |

|    |  |   |
|----|--|---|
| 39 | array_diff_assoc<br>(\$array1, \$array2)         | Trả về một mảng bao gồm các phần tử có khóa tồn tại trong mảng \$array1 nhưng không tồn tại trong mảng \$array2                             |
| 40 | array_intersect<br>(\$array1, \$array2)          | Trả về một mảng bao gồm các phần tử giống nhau về giá trị giữa 2 mảng \$array1 và \$array2  |
| 41 | array_intersect_key<br>(\$array1, \$array2)      | Trả về một mảng bao gồm các phần tử giống nhau về khóa giữa 2 mảng \$array1 và \$array2   |
| 42 | array_intersect_assoc<br>(\$array1, \$array2)    | Trả về một mảng bao gồm các phần tử giống nhau về khóa và giá trị giữa 2 mảng \$array1 và \$array2  |
| 43 | array_walk ()                                    | Gửi các giá trị của mảng đến một hàm nào đó để xử lý và nhận kết quả trả về là một mảng mới   |
| 44 | array_map ()                                     | Gửi các giá trị của một hay nhiều mảng đến một hàm nào đó để xử lý và nhận kết quả trả về là một mảng mới                                   |
| 45 | array_slice<br>(array, offset ,length,preserve)  | Trích xuất lấy một đoạn phần tử của mảng từ từ vị trí bắt đầu offset (vị trí bắt đầu trong mảng là 0) và lấy length phần tử.                |
| 46 | array_splice<br>(array1, offset ,length, array2) | Xóa bỏ một đoạn phần tử của mảng array1 từ từ vị trí bắt đầu và lấy length phần tử. Sau đó thay thế các phần tử bị loại bỏ bằng mảng array2 |
| 47 | sort(array)                                      | Sắp xếp các phần tử trong mảng array tăng dần theo giá trị  |
| 48 | rsort(array)                                     | Sắp xếp các phần tử trong mảng array giảm dần theo giá trị  |
| 49 | ksort(array)                                     | Sắp xếp các phần tử trong mảng array tăng dần theo khóa   |
| 50 | krsort(array)                                    | Sắp xếp các phần tử trong mảng array giảm dần theo khóa   |

## KỸ THUẬT XỬ LÝ CHUỖI

| STT | Hàm   | Chức năng  |
|-----|---|--|
| 1   | strlen()  | Lấy chiều dài của chuỗi (tổng số ký tự có trong chuỗi)   |
| 2   | mb_strlen()   | Lấy chiều dài của chuỗi UTF-8  |
| 3   | str_word_count()                                      | Đếm số từ có trong chuỗi   |
| 4   | strtoupper(\$str)                                     | Chuyển đổi chữ thường thành chữ HOA  |
| 5   | strtolower(\$str)                                     | Chuyển đổi chữ HOA thành chữ thường  |
| 6   | ucfirst(\$str)  | Chuyển đổi ký tự đầu tiên đầu tiên trong chuỗi thành chữ hoa   |
| 7   | lcfirst(\$str)  | Chuyển đổi ký tự đầu tiên đầu tiên trong chuỗi thành chữ thường  |
| 8   | ucwords(\$str)  | Chuyển đổi tất cả các ký tự đầu tiên của các từ trong một chuỗi thành chữ in hoa   |
| 9   | stripos()   | Tìm kiếm chỉ số xuất hiện đầu tiên của một từ nào đó trong chuỗi   |
| 10  | strripos()  | Tìm kiếm chỉ số xuất hiện cuối cùng của một từ nào đó trong chuỗi  |
| 11  | strrev()  | Đảo ngược một chuỗi  |
| 12  | substr()  | Trích xuất nội dung nào đó trong chuỗi   |
| 13  | ltrim(\$str, \$params)                                | Xóa các ký nằm bên trái của một chuỗi nào đó   |
| 14  | rtrim(\$str, \$params)                                | Xóa các ký nằm bên phải của một chuỗi nào đó   |
| 15  | trim(\$str, \$params)                                 | Xóa các ký nằm bên phải và bên trái của một chuỗi nào đó   |
| 16  | implode (\$str, \$array)                              | Chuyển các giá trị của mảng \$array thành một chuỗi bao gồm các phần tử cách nhau bởi ký tự \$str                          |
| 17  | explode (\$delimiter, \$str)                          | Chuyển một chuỗi thành một mảng. Tách chuỗi dựa vào \$delimiter, mỗi đoàn tách ra sẽ thành một phần tử của mảng mới        |
| 18  | str_repeat(\$str,n)                                   | Lặp lại chuỗi \$str với số lần lặp là n  |
| 19  | chr()   | Trả về ký tự tương ứng với mã ASCII được truyền vào  |
| 20  | ord()   | Trả về giá trị ASCII của ký tự đầu tiên trong chuỗi  |
| 21  | parse_str()   | Chuyển các nội dung truy vấn vào các biến hoặc mảng  |
| 22  | parse_url()   | Truy xuất các thành phần protocol, domain name, path, .. của một URL nào đó  |
| 23  | strcmp(\$str1, \$str2)                                | So sánh hai chuỗi \$str1 và \$str2 với nhau  |
| 24  | substr_compare<br>(\$str1, \$str2, \$start, \$length) | Lấy \$length phần tử từ vị trí \$start trong chuỗi \$str1 say đó so sánh với chuỗi \$str2                                  |
| 25  | str_pad<br>(\$str, \$length, \$padString, \$padType)  | Tăng độ dài của chuỗi \$str thành \$length với các ký tự mới được thêm vào là \$pad_string (với cơ chế thêm là \$pad_type) |

|    |   |  |
|----|---|--|
| 26 | <code>str_shuffle()</code>  | Sắp xếp ngẫu nhiên thứ tự các ký tự trong chuỗi  |
| 27 | <code>str_replace(\$find, \$replace, \$string)</code>               | Tìm kiếm và thay thế giá trị \$find trong chuỗi \$string bằng giá trị \$replace  |
| 28 | <code>substr_count(\$string, \$substring, \$start, \$length)</code> | Lấy \$length phần tử từ vị trí \$start trong chuỗi \$str và thống kê số lần xuất hiện của \$substring trong chuỗi vừa lấy trên |
| 29 | <code>str_split(\$str, \$length)</code>                             | Cắt chuỗi thành từng phần tử trong mảng, mỗi phần tử có độ dài là \$length ký tự   |
| 30 | <code>addslashes(\$str)</code>                                      | Thêm ký tự \ vào trước các ký tự: nháy đơn ('), nháy đôi ("), gạch chéo (\) và NULL  |
| 31 | <code>addcslashes(\$str, \$character)</code>                        | Thêm ký tự \ vào trước ký tự \$character   |
| 32 | <code>stripslashes (\$str, \$character)</code>                      | Hiển thị chuỗi không có các ký tự gạch chéo được tạo bởi hàm addslashes  |
| 33 | <code>stripcslashes (\$str, \$character)</code>                     | Hiển thị chuỗi không có các ký tự gạch chéo được tạo bởi hàm addcslashes   |
| 34 | <code>htmlspecialchars (\$str)</code>                               | Chuyển đổi các ký tự được quy định trước & ' < > sang giá trị HTML entities  |
| 35 | <code>htmlspecialchars_decode (\$str)</code>                        | Chuyển đổi các giá trị HTML entities được gọi bởi hàm htmlspecialchars (\$str) về giá trị ban đầu                              |
| 36 | <code>htmlentities(\$str)</code>                                    | Chuyển đổi các ký tự sang giá trị HTML entities  |
| 37 | <code>html_entity_decode(\$str)</code>                              | Chuyển đổi các giá trị HTML entities được gọi bởi hàm htmlentities(\$str) về giá trị ban đầu                                   |
| 38 | <code>get_html_translation_table()</code>                           | Xem danh sách các giá trị HTML entities  |
| 39 | <code>strip_tags()</code>   | Loại bỏ các thẻ HTML   |

## LÀM VIỆC VỚI NUMBER

| STT | Hàm                              | Chức năng  |
|-----|----------------------------------|--|
| 1   | is_numeric()                     | Kiểm tra biến có lưu giá trị kiểu Number hay không             |
| 2   | is_int                           | Kiểm tra biến có lưu giá trị kiểu Integer hay không            |
| 3   | is_float                         | Kiểm tra biến có lưu giá trị kiểu Float hay không              |
| 4   | range(\$start, \$length, \$loop) | Tạo ra một dãy số với số bắt đầu và số kết thúc được cho trước |
| 5   | round()                          | Làm tròn đến số nguyên gần nhất                                |
| 6   | ceil()                           | Làm tròn đến số nguyên gần nhất và lớn nhất                    |
| 7   | floor()                          | Làm tròn đến số nguyên gần nhất và nhỏ nhất                    |
| 8   | min()                            | Lấy giá trị nhỏ nhất trong các số được truyền vào              |
| 9   | max()                            | Lấy giá trị lớn nhất trong các số được truyền vào              |
| 10  | rand(min, max)                   | Lấy giá trị ngẫu nhiên được trả về nằm trong đoạn [min,max]    |
| 11  | number_format()                  | Định dạng cách hiển thị giữa các phần nghìn trong 1 số         |
| 12  | abs(\$number)                    | Trả về giá trị tuyệt đối của một số nào đó                     |
| 13  | pow(x, y)                        | Trả về kết quả là x mũ y                                       |
| 14  | sqrt(\$number)                   | Tính căn bậc hai của \$number                                  |

## XỬ LÝ THỜI GIAN

| STT | Hàm                                      | Chức năng   |
|-----|--|---|
| 1   | getdate()                                | Lấy thời gian hiện tại được thiết lập ở máy chủ                             |
| 2   | date_default_timezone_get()              | Trả về kết quả múi giờ đã được thiết lập trước                              |
| 3   | date_default_timezone_set                | Thiết lập múi giờ   |
| 4   | time()                                   | Trả về số giây từ thời điểm hiện tại so với 01/01/1970                      |
| 5   | mktime()                                 | Trả về số giây tại một thời điểm nào đó so với 01/01/1970                   |
| 6   | date()                                   | Định dạng cách hiển thị thời gian   |
| 7   | checkdate(month, day, year)              | Kiểm tra ngày hợp lệ  |
| 8   | strtotime()                              | Chuyển đổi chuỗi thời gian (ở định dạng English) về giá trị timestamp       |
| 9   | date_parse_from_format(\$format, \$date) | Chuyển đổi chuỗi thời gian \$date về mảng thời gian theo định dạng \$format |

## XỬ LÝ THỜI GIAN TRONG MYSQL

Trong nội dung này, tập trung trình bày các hàm xử lý thời gian, ngày, tháng, năm, ... các thao công trừ và định dạng thời gian trong hệ quản trị cơ sở dữ liệu MySQL.

### 1. NOW()

- Xác định thời gian tại thời điểm **hiện tại**

| STT | Đầu vào           | Đầu ra              |
|-----|-------------------|---------------------|
| 1   | SELECT NOW();     | 2011-12-26 19:50:24 |
| 2   | SELECT NOW() + 0; | 20111226195113      |

### 2. CURDATE()

- Xác định ngày, tháng, năm tại thời điểm **hiện tại**

| STT | Đầu vào           | Đầu ra     |
|-----|-------------------|------------|
| 1   | SELECT CURDATE(); | 2011-12-26 |

### 3. CURTIME()

- Xác định giờ, phút, giây tại thời điểm **hiện tại**

| STT | Đầu vào           | Đầu ra   |
|-----|-------------------|----------|
| 1   | SELECT CURTIME(); | 19:54:27 |

### 4. DATE(expr)

- Xác định ngày, tháng, năm tại thời điểm **expr**

| STT | Đầu vào                             | Đầu ra     |
|-----|-------------------------------------|------------|
| 1   | SELECT DATE('2011-12-23 09:22:03'); | 2011-12-23 |

### 5. TIME(expr)

- Xác định giờ, phút, giây tại thời điểm **expr**

| STT | Đầu vào                             | Đầu ra   |
|-----|-------------------------------------|----------|
| 1   | SELECT TIME('2011-12-23 09:22:03'); | 09:22:03 |

### 6. YEAR(expr)

- Xác định **năm** tại thời điểm **expr**

| STT | Đầu vào                    | Đầu ra |
|-----|----------------------------|--------|
| 1   | SELECT YEAR('2011-12-23'); | 2011   |

## 7. MONTH(expr)

- Xác định **tháng** (chỉ số 1, 2, 3, ...) tại thời điểm **expr**

| STT | Đầu vào                     | Đầu ra |
|-----|-----------------------------|--------|
| 1   | SELECT MONTH('2011-12-23'); | 12     |

## 8. MONTHNAME(expr)

- Xác định **tên tháng** tại thời điểm **expr**

| STT | Đầu vào                         | Đầu ra   |
|-----|---------------------------------|----------|
| 1   | SELECT MONTHNAME('2011-12-23'); | December |

## 9. WEEKDAY(expr)

- Xác định **chỉ số** ngày trong **tuần** (0 = Monday, 1 = Tuesday, ... 6 = Sunday)

| STT | Đầu vào                       | Đầu ra |
|-----|-------------------------------|--------|
| 1   | SELECT WEEKDAY('2011-12-23'); | 4      |

## 10. WEEKOFYEAR(expr)

- Xác định **chỉ số** tuần trong **năm** (kết quả nằm trong khoảng từ 1 đến 55)

| STT | Đầu vào                          | Đầu ra |
|-----|----------------------------------|--------|
| 1   | SELECT WEEKOFYEAR('2011-12-23'); | 51     |

## 11. DAY(expr)

- Xác định **ngày** (**chỉ số**) tại thời điểm **expr**

| STT | Đầu vào                   | Đầu ra |
|-----|---------------------------|--------|
| 1   | SELECT DAY('2011-12-23'); | 23     |

## 12. DAYOFYEAR(expr)

- Xác định **số thứ tự** của ngày trong **năm** tại thời điểm **expr** (kết quả nằm trong khoảng từ 1 đến 366)

| STT | Đầu vào                         | Đầu ra |
|-----|---------------------------------|--------|
| 1   | SELECT DAYOFYEAR('2011-12-23'); | 357    |

## 13. DAYOFMONTH(expr)

- Xác định **số thứ tự** của ngày trong **tháng** tại thời điểm **expr** (kết quả nằm trong khoảng từ 1 đến 31)

| STT | Đầu vào                          | Đầu ra |
|-----|----------------------------------|--------|
| 1   | SELECT DAYOFMONTH('2011-12-23'); | 23     |

#### 14. DAYNAME(expr)

- Xác định **tên (thứ)** của ngày tại thời điểm **expr**

| STT | Đầu vào                       | Đầu ra |
|-----|-------------------------------|--------|
| 1   | SELECT DAYNAME('2011-12-23'); | Friday |

#### 15. DAYOFWEEK(expr)

- Xác định **ngày (chỉ số)** của ngày tại thời điểm **expr** (1 = Sunday, 2 = Monday, ..., 7 = Saturday)

| STT | Đầu vào                         | Đầu ra |
|-----|---------------------------------|--------|
| 1   | SELECT DAYOFWEEK('2011-12-23'); | 6      |

#### 16. HOUR(expr)

- Xác định **giờ** tại thời điểm **expr**

| STT | Đầu vào                             | Đầu ra |
|-----|-------------------------------------|--------|
| 1   | SELECT HOUR('2011-12-23 09:22:03'); | 9      |

#### 17. MINUTE(expr)

- Xác định **phút** tại thời điểm **expr**

| STT | Đầu vào                               | Đầu ra |
|-----|---------------------------------------|--------|
| 1   | SELECT MINUTE('2011-12-23 09:22:03'); | 22     |

#### 18. SECOND(expr)

- Xác định **giây** tại thời điểm **expr**

| STT | Đầu vào                               | Đầu ra |
|-----|---------------------------------------|--------|
| 1   | SELECT SECOND('2011-12-23 09:22:03'); | 3      |

#### 19. MICROSECOND(expr)

- Xác định **mili giây** tại thời điểm **expr**

| STT | Đầu vào                                | Đầu ra |
|-----|--|--------|
| 1   | SELECT MICROSECOND('12:00:00.123456'); | 123456 |

#### 20. TIME\_TO\_SEC(expr)

- Chuyển thời gian tại thời điểm **expr** ra số giây

| STT | Đầu vào                         | Đầu ra |
|-----|---------------------------------|--------|
| 1   | SELECT TIME_TO_SEC('22:23:00'); | 80580  |

## 21. FROM\_DAYS(N)

- Chuyển về định dạng thời gian của giá trị số N

| STT | Đầu vào                   | Đầu ra     |
|-----|---------------------------|------------|
| 1   | SELECT FROM_DAYS(730669); | 2000-07-03 |

## 22. LAST\_DAY(expr)

- Xác định **ngày cuối cùng** trong tháng của tháng tại thời điểm expr

| STT | Đầu vào                                 | Đầu ra     |
|-----|---|------------|
| 1   | SELECT LAST_DAY('2011-12-23 09:22:03'); | 2011-12-31 |

## 23. ADDDATE(date, INTERVAL expr unit)

- Cộng vào thời điểm date một giá trị expr thuộc dạng unit

| STT | Đầu vào   | Đầu ra     |
|-----|---|------------|
| 1   | SELECT ADDDATE('2006-05-12', INTERVAL 1 DAY);   | 2006-05-13 |
| 2   | SELECT ADDDATE('2006-05-12', INTERVAL 8 MONTH); | 2007-01-12 |

### Bảng định dạng và kiểu của các thuộc tính expr và unit

| unit               | expr                                      |
|--------------------|---|
| MICROSECOND        | MICROSECONDS                              |
| SECOND             | SECONDS                                   |
| MINUTE             | MINUTES                                   |
| HOUR               | HOURS                                     |
| DAY                | DAYS                                      |
| WEEK               | WEEKS                                     |
| MONTH              | MONTHS                                    |
| QUARTER            | QUARTERS                                  |
| YEAR               | YEARS                                     |
| SECOND_MICROSECOND | 'SECONDS.MICROSECONDS'                    |
| MINUTE_MICROSECOND | 'MINUTES:SECONDS.MICROSECONDS'            |
| MINUTE_SECOND      | 'MINUTES:SECONDS'                         |
| HOUR_MICROSECOND   | 'HOURS:MINUTES:SECONDS.MICROSECONDS'      |
| HOUR_SECOND        | 'HOURS:MINUTES:SECONDS'                   |
| HOUR_MINUTE        | 'HOURS:MINUTES'                           |
| DAY_MICROSECOND    | 'DAYS HOURS:MINUTES:SECONDS.MICROSECONDS' |
| DAY_SECOND         | 'DAYS HOURS:MINUTES:SECONDS'              |
| DAY_MINUTE         | 'DAYS HOURS:MINUTES'                      |
| DAY_HOUR           | 'DAYS HOURS'                              |
| YEAR_MONTH         | 'YEARS-MONTHS'                            |

## 24. DATE\_ADD(date, INTERVAL expr unit)

- Cộng vào thời điểm date một giá trị expr thuộc dạng unit

| STT | Đầu vào  | Đầu ra     |
|-----|--|------------|
| 1   | SELECT DATE_ADD('2006-05-12', INTERVAL 1 MONTH); | 2006-06-12 |

## 25. DATEDIFF(expr1,expr2)

- Tính số ngày giữa hai khoảng thời gian cụ thể (chỉ phần ngày, tháng, năm của hai khoảng thời gian nay mới được tính toán)

| STT | Đầu vào   | Đầu ra |
|-----|---|--------|
| 1   | SELECT DATEDIFF('2011-12-30', '2011-12-23 09:22:03'); | 7      |

## 26. SUBDATE(date, INTERVAL expr unit)

- Trừ vào thời điểm **date** một giá trị **expr** thuộc dạng **unit**

| STT | Đầu vào                                       | Đầu ra     |
|-----|---|------------|
| 1   | SELECT SUBDATE('2006-05-12', INTERVAL 1 DAY); | 2006-05-11 |

## 27. DATE\_SUB(date, INTERVAL expr unit)

- Trừ vào thời điểm **date** một giá trị **expr** thuộc dạng **unit**

| STT | Đầu vào   | Đầu ra     |
|-----|---|------------|
| 1   | SELECT DATE_SUB ('2006-05-12', INTERVAL 1 MONTH); | 2006-04-12 |

## 28. SUBTIME(expr1,expr2)

- Xác định hiệu của **expr1** và **expr2**, giá trị kết quả này có định dạng như **expr1**

| STT | Đầu vào  | Đầu ra                        |
|-----|--|-------------------------------|
| 1   | SELECT SUBTIME('2007-12-31 23:59:59.999999','1 1:1:1.000002'); | 2007-12-30<br>22:58:58.999997 |

## 29. EXTRACT(unit FROM date)

- Xác định chính xác thành phần **unit** tại thời điểm date

| STT | Đầu vào  | Đầu ra |
|-----|--|--------|
| 1   | SELECT EXTRACT(YEAR FROM '2011-12-23');                | 2011   |
| 2   | SELECT EXTRACT(YEAR_MONTH FROM '2011-12-23 01:02:03'); | 201112 |

## 30. DATE\_FORMAT(date,format)

- Định dạng thời gian tại thời điểm **date** với kiểu định dạng **format**.

**Bảng các ký hiệu định dạng:**

| Ký hiệu | Ý nghĩa   |
|---------|---|
| %a      | Tên viết tắt các ngày trong tuần (Sun, ..., Sat )                                       |
| %b      | Tên viết tắt các tháng trong năm (Jan, ..., Dec )                                       |
| %c      | Tháng (0,1, ..., 12)  |
| %D      | Thứ tự các ngày trong tháng (0 <sup>th</sup> , 1 <sup>st</sup> , 2 <sup>nd</sup> , ...) |
| %d      | Thứ tự các ngày trong tháng (00, 01, ..., 31)   |
| %e      | Thứ tự các ngày trong tháng (0, 1, ..., 31)   |
| %f      | Microseconds (000000..999999)   |
| %H      | Giờ (00..23)  |
| %h      | Giờ (01..12)  |
| %I      | Giờ (01..12)  |
| %i      | Phút (00..59)   |
| %j      | Số thứ tự ngày trong năm (001..366)   |
| %k      | Giờ (0..23)   |
| %l      | Giờ (1..12)   |
| %M      | Tên tháng (January..December)   |
| %m      | Tháng (00..12)  |
| %p      | AM hoặc PM  |
| %r      | Giờ (00..12) (hh:mm:ss theo sau bởi AM hoặc PM)   |
| %S      | Giây (00..59)   |
| %s      | Giây (00..59)   |
| %T      | Giờ (00..23) (hh:mm:ss)   |
| %U      | Số thứ tự tuần trong năm (00..53), khi chủ nhật là ngày đầu tiên của tuần               |
| %u      | Số thứ tự tuần trong năm (00..53), khi thứ hai là ngày đầu tiên của tuần                |
| %w      | Số thứ tự của ngày trong tuần (0=Sunday..6=Saturday)                                    |

| STT | Đầu vào  | Đầu ra                |
|-----|--|-----------------------|
| 1   | SELECT DATE_FORMAT('2009-10-04 22:23:00', '%W %M %Y'); | 'Sunday October 2009' |
| 2   | SELECT DATE_FORMAT('2007-10-04 22:23:00', '%H:%i:%s'); | '22:23:00'            |
| 3   | SELECT DATE_FORMAT('1999-01-01', '%X %V');             | '1998 52'             |
| 4   | SELECT DATE_FORMAT('2006-06-01', '%d');                | '01'                  |

### 31. GET\_FORMAT({DATE|TIME|DATETIME}, {'EUR'|'USA'|'JIS'|'ISO'|'INTERNAL'})

- Xác định định dạng thời gian theo từng vùng. Hàm này thường đường sử dụng kết hợp với hàm DATE\_FORMAT

| STT | Đầu vào   | Đầu ra     |
|-----|---|------------|
| 1   | SELECT DATE_FORMAT('2003-10-03',GET_FORMAT(DATE,'EUR'));  | 03.10.2003 |
| 2   | SELECT STR_TO_DATE('10.31.2003', GET_FORMAT(DATE,'USA')); | 2003-10-31 |

# XỬ LÝ CHUỖI TRONG MYSQL

Trong nội dung này, tập trung trình bày các hàm xử lý chuỗi, các thao tác tìm kiếm chuỗi cơ bản kết hợp với việc sử dụng biểu thức chính quy trong hệ quản trị cơ sở dữ liệu MySQL.

## A. CÁC HÀM XỬ LÝ CHUỖI CƠ BẢN:

### 1. LENGTH (str):

- Xác định chiều dài chuỗi str

| STT | Đầu vào                          | Đầu ra |
|-----|----------------------------------|--------|
| 1   | SELECT LENGTH ('MySQL is easy'); | 13     |

### 2. CONCAT (str1, str2, ..., strn)

- Nối các chuỗi str1, str2, ..., strn thành một chuỗi duy nhất

| STT | Đầu vào  | Đầu ra        |
|-----|--|---------------|
| 1   | SELECT CONCAT ('MySQL', ' is', ' easy');       | MySQL is easy |
| 2   | SELECT CONCAT ('MySQL', ' is', NULL, ' easy'); | NULL          |

### 3. FORMAT (str, n)

- Chuyển đổi định dạng chuỗi str theo dạng #,###,###.##, trong đó làm tròn đến n số thập phân.

| STT | Đầu vào                          | Đầu ra      |
|-----|----------------------------------|-------------|
| 1   | SELECT FORMAT (12345.678901, 4); | 12,345.6789 |
| 2   | SELECT FORMAT (12345.54321, 0);  | 12,346      |

### 4. INSERT (str, pos, len, newstr)

- Thay thế các ký tự trong chuỗi str từ vị trí thứ pos đến vị trí thứ pos+len-1 bởi chuỗi ký tự newstr.

| STT | Đầu vào   | Đầu ra            |
|-----|---|-------------------|
| 1   | SELECT INSERT ('MySQL is easy', 3, 3, 'Library');   | MyLibrary is easy |
| 2   | SELECT INSERT ('MySQL is easy', 3, 100, 'Library'); | MyLibrary         |

### 5. INSTR (str, substr)

- Trả về vị trí xuất hiện đầu tiên của chuỗi substr trong chuỗi str

| STT | Đầu vào                           | Đầu ra |
|-----|-----------------------------------|--------|
| 1   | SELECT INSTR ('MySQL', 'SQL');    | 3      |
| 2   | SELECT INSTR ('MySQL', 'Server'); | 0      |

## 6. LEFT (str, len)

- Trả về chuỗi ký tự tính từ vị trí bên trái của chuỗi **str** từ vị trí thứ nhất đến vị trí thứ **len**.

| STT | Đầu vào                           | Đầu ra |
|-----|-----------------------------------|--------|
| 1   | SELECT LEFT ('MySQL is easy', 5); | MySQL  |

## 7. RIGHT (str, len)

- Trả về chuỗi ký tự tính từ vị trí bên phải của chuỗi **str** từ vị trí thứ nhất đến vị trí thứ **len**.

| STT | Đầu vào                            | Đầu ra  |
|-----|------------------------------------|---------|
| 1   | SELECT RIGHT ('MySQL is easy', 7); | is easy |

## 8. LOCATE (substr, str, pos)

- Trả về vị trí xuất hiện đầu tiên của chuỗi **substr** trong chuỗi **str** tính từ vị trí thứ **pos**

| STT | Đầu vào  | Đầu ra |
|-----|--|--------|
| 1   | SELECT LOCATE ('very', 'MySQL is very very easy');     | 10     |
| 2   | SELECT LOCATE ('very', 'MySQL is very very easy', 11); | 15     |

## 9. LOWER (str)

- Chuyển đổi chuỗi **str** thành chữ thường

| STT | Đầu vào                         | Đầu ra        |
|-----|---------------------------------|---------------|
| 1   | SELECT LOWER ('MySQL is easy'); | mysql is easy |

## 10. UPPER (str)

- Chuyển đổi chuỗi **str** thành chữ hoa

| STT | Đầu vào                         | Đầu ra        |
|-----|---------------------------------|---------------|
| 1   | SELECT UPPER ('MySQL is easy'); | MYSQL IS EASY |

## 11. LTRIM (str)

- Loại bỏ tất cả khoảng trắng bên trái chuỗi **str**

| STT | Đầu vào                         | Đầu ra        |
|-----|---------------------------------|---------------|
| 1   | SELECT LTRIM(' MySQL is easy'); | MySQL is easy |

## 12. RTRIM (str)

- Loại bỏ tất cả khoảng trắng bên phải chuỗi **str**

| STT | Đầu vào                         | Đầu ra        |
|-----|---------------------------------|---------------|
| 1   | SELECT RTRIM('MySQL is easy '); | MySQL is easy |

### 13. REPLACE (str, from\_str, to\_str)

- Thay thế các ký tự **from\_str** trong chuỗi **str** bởi các ký tự **to\_str**

| STT | Đầu vào   | Đầu ra      |
|-----|---|-------------|
| 1   | SELECT REPLACE ('MySQL is easy', 'MySQL', 'PHP'); | PHP is easy |

### 14. REVERSE (str)

- Đảo ngược chuỗi str

| STT | Đầu vào   | Đầu ra      |
|-----|---|-------------|
| 1   | SELECT REPLACE ('MySQL is easy', 'MySQL', 'PHP'); | PHP is easy |

### 15. SUBSTRING (str FROM pos FOR len)

- Trích lọc từ chuỗi str một chuỗi mới, chuỗi mới này có các ký tự bắt đầu từ vị trí **pos** (tính từ vị trí biên trái) cho đến vị trí **pos+len-1**.

| STT | Đầu vào  | Đầu ra    |
|-----|--|-----------|
| 1   | SELECT SUBSTRING('MySQL is easy' FROM 5 FOR 10); | L is easy |
| 2   | SELECT SUBSTRING('MySQL is easy' FROM -4 FOR 2); | ea        |

### 16. TRIM ([{BOTH | LEADING | TRAILING} [remstr] FROM] str), TRIM([remstr FROM] str)

- Trả về chuỗi str với các ký tự **remstr** đã bị loại bỏ, theo các tiêu chí **LEADING** (bên trái), **TRAILING** (bên phải) và **BOTH** (cả hai vị trí bên trái và bên phải).

| STT | Đầu vào  | Đầu ra             |
|-----|--|--------------------|
| 1   | SELECT TRIM (' MySQL is easy '); -- MySQL is easy        | MySQL is easy      |
| 2   | SELECT TRIM (LEADING '-' FROM '---MySQL is easy-----');  | MySQL is easy----- |
| 3   | SELECT TRIM (TRAILING '-' FROM '---MySQL is easy-----'); | ---MySQL is easy   |
| 4   | SELECT TRIM (BOTH '-' FROM '---MySQL is easy-----');     | MySQL is easy      |

## B. KẾT HỢP BIỂU THỨC CHÍNH QUY:

- Biểu thức chính quy được sử dụng khá thường xuyên và đây là một phương pháp tương đối hiệu quả cho các công việc tìm kiếm phức tạp.
- Sử dụng biểu thức chính quy trong MySQL được chia thành 3 nhóm từ khóa sau:
  - o NOT\_REGEXP
  - o REGEXP
  - o RLIKE
- Trong nội dung này trình bày việc sử dụng biểu thức chính quy với từ khóa **REGEXP**. Kết quả tìm kiếm sẽ trả về 1 nếu tìm thấy hoặc trả về 0 nếu không tìm thấy.

| TT | Ký hiệu      | Diễn giải  | Ví dụ                                   |        |
|----|--------------|--|---|--------|
|    |              |  | Đầu vào                                 | Đầu ra |
| 1  | <b>^</b>     | Tìm từ đầu chuỗi nguồn   | SELECT 'MySQL is easy' REGEXP '^MyN';   | 0      |
| 2  | <b>\$</b>    | Tìm từ cuối chuỗi nguồn  | SELECT 'MySQL is easy' REGEXP 'easy\$'; | 1      |
| 3  | <b>.</b>     | Đại diện một ký tự bất kỳ  | SELECT 'MySQL' REGEXP '^M.SQL\$';       | 1      |
| 4  | <b>*</b>     | Ký tự xuất hiện 0 hoặc nhiều lần   | SELECT 'MyyySQL' REGEXP '^M.*SQL\$';    | 1      |
| 5  | <b>+</b>     | Ký tự xuất hiện 1 hoặc nhiều lần   | SELECT 'MSQL' REGEXP '^M.+SQL\$';       | 0      |
| 6  | <b>?</b>     | Ký tự xuất hiện 0 hoặc 1 lần   | SELECT 'MyyySQL' REGEXP '^M.?SQL\$';    | 0      |
| 7  | <b> </b>     | Sự lựa chọn (hoặc)   | SELECT 'My' REGEXP '^^(My Me)\$';       | 1      |
| 8  | <b>{n.m}</b> | Số lần xuất hiện của ký tự (a* tương đương a{0}; a+ tương đương a{1}; a? tương đương a{0,1}) | SELECT 'abcde' REGEXP 'a[bcd]{1,10}e';  | 1      |